- BACHELOR THESIS -

COVID-19 impact research by analysing data and patterns of events gathered from GDELT project

ERASMUS+

Semester: Spring 2020

Bachelor's degree: Network Engineering

Author: Reva Dhawan Soni

Supervisors: Jarosław Koźlak (AGH), Enrica Zola (UPC)

Date: 14.09.2020

AKADEMIA GÓRNICZO-HUTNICZA
FACULTY OF COMPUTER SCIENCE,
ELECTRONICS AND TELECOMMUNICATIONS
DEPT. OF COMPUTER SCIENCE9

UPC UNIVERSITY
SCHOOL OF TELECOMMUNICATIONS
AND AEROSPACE ENGINEERING

ACKNOWLEDGEMENTS

I would first like to thank my supervisor Prof. Jarosław Koźlak from the Faculty of Computer Science at AGH University of Science and Technology. First, for giving me the opportunity to realize this project during my semester in Krakow, Poland. Also, he has been by my side with all the development of my thesis by cooperating, advising and guiding me whenever I was lost.

In addition, I would like to thank my supervisor Prof. Enrica Zola from the EETAC Faculty at UPC. She has been giving me support from the very first moment not only with this project but also with the whole Erasmus+ experience.

Finally, it is necessary for me to thank my mentor from ESN program, Sebastian Rolek. He supported me in all the difficult moments I had while realizing this project, giving me the self-confidence I needed in order to progress and bring out the best of me.

ABSTRACT

The GDELT Project [1] is a real-time network diagram and database of global human society for open research that captures what is happening around the world identifying the people, locations, organizations, themes, sources, emotions, counts, quotes, images and events driving our society every second of every day. It can help to analyse political events and searching for frequent patterns. The applied methods for the thesis are data mining, machine learning and complex network analysis. In other words, these methods consist on extracting information from a data set and transform it into a comprehensible structure for further use in order to describe the model and scenarios of experiments as well as their interpretation.

El Projecte GDELT [1] és un diagrama de xarxa i base de dades en temps real de la societat humana global per a investigacions obertes que recull el que està passant a tot el món identificant les persones, ubicacions, organitzacions, temes, fonts, emocions, recomptes, pressupostos, imatges i esdeveniments que condueixen la nostra societat cada segon del dia a dia. Pot ajudar a analitzar els esdeveniments polítics i cercar patrons freqüents tenint en compte aquests conjunts de dades. Els mètodes aplicats per a la tesi són la mineria de dades, l'aprenentatge de màquines i l'anàlisi complexa de xarxa. Dit d'una altra manera, aquests mètodes consisteixen en extreure informació d'un conjunt de dades i transformar-la en una estructura comprensible per a un ús posterior per tal de poder descriure el model i els escenaris dels experiments.

TABLE OF CONTENTS

Acronyms, Abbreviations and Initialisms

ADM1	First-order Administrative Division
AI	Artificial Intelligence
API	Application Programming Interface
CAMEO	Conflict and Mediation Event Observations
COVID-19	Coronavirus disease
CSV	Comma-separated values
ESN	Erasmus Student Network
GDELT	Global Database of Events, Language, and Tone
GDP	Gross Domestic Product
GNS	GeoNet Name Server
GNS/GNIS	GEOnet Names Server
IDE	Integrated Development Environment
IT	Information Technology
JSON	JavaScript Object Notation
NumPy	Numerical Python
Spyder	Scientific Python Development Environment
UK	United Kingdom
US/USA	United States of America

INTRODUCTION

Thesis motivation

The GDELT Project [1] is the largest open database ever created that gathers the human society behavior such as political meetings, conflicts, protests, countries providing aid, etc. The simple fact of being able to use it for any kind of application makes it even more attractive and challenging. For instance, this information based on international events can be used for watch boarding, forecasting, early warning, alert services, etc. Therefore, the motivation of the thesis is to be capable of using this huge amount of information in the database by applying methods previously studied during my Bachelor's degree in order to demonstrate how we can use this engineering approach in very different fields. And from another point of view, there is a huge motivation to make a research of the very actual topic of Coronavirus pandemic with the information provided. It has been a very important and problematic worldwide fact which will mark the before and after of our lives and that is the main reason why it is a very interesting subject to analyze.

Objectives

From a general point of view, the main objective consists on the analysis in time of the datasets extracted from the GDELT project [1]. In other words, what we want is to be able to identify groups and patterns considering worldwide political events during the COVID-19 pandemic through the analysis of dynamic data sets gathered from GDELT project. To make it possible, we build a system structure that focuses on analyzing different aspects of the data such as the evolution in time of political events, incidents or social occurrences during the first half of the year 2020, by considering the virus. Finally, through this study we will provide some important facts such as why the pandemic had a different impact depending on the continent or if it produced positive consequences. To

sum up, with this research we will be able to show how much knowledge we can get thanks to data analysis.

Document structure

This project is structured in nine chapters. The first one is the introduction which contains the motivation, the objectives and the structure of the thesis. The second chapter corresponds to the theoretical background on which the thesis is based. We will provide information about the database used and some taxonomies. It is crucial to understand how the database is structured and what type of information it can provide from a theoretical point of view.

Chapter 6 is about the technologies and the applied methods. The three main techniques used are data mining, machine learning and complex network analysis. Moreover, we are going to dig into machine learning by using two types of algorithms that can give us useful information.

Then, once we know the basis, we can introduce the main idea of this thesis. It explains more specifically what is going to be done in the practical part of the project such as useful measures that are going to be used, which type of analysis, etc. Therefore, the fourth chapter contains the development and the implementation of this idea. It explains how to prepare the dataset and the environment structure in order to make this development work.

Consequently, the chapters from five to seven represent three different parts of the entire analysis. The first one describes general aspects such as the quantity of events per month, their evolution in time and which type of events (political, protests, aids…) are more frequent. Moreover, there is a small subchapter about China and its correlation with the rest of countries as we all know that the COVID-19 started there.

Chapter 6 is about clustering, a machine learning method that is very useful to find groups and patterns among all the dataset. It gives us information about the

evolution of events, whether their behaviour got better or not by considering measures such as the impact on the stability of the country.
The seventh chapter represents different types of analysis related to countries. It includes general analysis like the most popular countries per month, world maps that show the behaviour of each region and also, the analysis of different types of countries clustering.

Then, the discussion of these results is provided in chapter 8. It sums up the useful information extracted from the previous three chapters. For example, one interesting fact is how the density of different types of events changed. For instance, at the beginning of the year there were more negative events such as conflicts or protests. However, during the lockdown the impact of the pandemic changed these values. Also, it concludes the stability of countries; rather if their behaviour is connected to the spread of the virus inside their own borders or not.

And finally, the last chapter contains the conclusions, the future work and some perspectives we may have related with this project. It concludes the overall experience while realizing this project, some summary about the results and finally what this project meant to me. In addition, the future work represents the performance that could have been done in order to extract even more information. The data analysis world can be infinite.

1. THEORETICAL BACKGROUND

In order to understand what this project is about it is necessary to know what we are basing it on. Therefore, in the theoretical background it is going to be explained what is the database “GDELT project” [1] and its ontology called “CAMEO” [2].

1.1. GDELT Project

The GDELT Project, or *Global Database of Events, Language and Tone* [1], is a real-time network diagram and database of global human society for open research that captures what is happening around the world, what its context is and who’s involved, and how the world is feeling about it, every single day.

In other words, it is the largest, most comprehensive and highest resolution open database of human society ever created. It was developed by *Kalev Leetaru* in 2011 and it includes data from 1979 to the present. It came from a desire to understand in a better way the human society and especially the connection between communicative discourse and physical societal-scale behavior. Therefore, we can say that the aim is to codify the entire planet into a computable format using all available open information sources in order to understand the global world.

Finally, all the data are available for download via zip files and, since 2014, is query-able via Google’s BigQuery web interface and through its API, and with the GDELT Analysis Service. These two concepts are going to be described later in order to understand how all this database works.

1.1.1. Features

Global Reach

It monitors the world's broadcast such as press, print, and web news from across every country in the world in over 100 languages to keep continually updated on breaking developments anywhere on the planet.

Emerging Media

It explores how social media is used around the world and how people and societies express themselves and talk about the world online.

Historical Breadth

GDELT is the first truly multi-decade global event database and through an array of collaborations and partnerships it is expanding its coverage all the way back to the year 1800.

Translation

The GDELT Translingual platform represents the largest real-time streaming news machine translation deployment in the world as it monitors in 65 languages and this non-English volume is translated in real-time into English and proceed.

1.1.2. Querying, Analyzing and Downloading

The entire GDELT database is totally free and open which means that everyone is able to download it, visualize it or analyze it at limitless scale. For that, there are three possible methods to use this vast archive of human society.

GDELT Analysis Service

It is a free cloud-based service that offers a variety of tools and services to allow you to visualize, explore and export not only the GDELT Event Database but also the GDELT Global Knowledge Graph. The second one attempt to connect every person, organization, location, news sources, etc. into a single massive network capturing what is happening around the world. It would be great for this project to apply the method of complex network analysis, which we are going to describe later.

To explain how it works, imagine that you want to export some particular event. First of all, it asks you for your Email address and for the date range which can be from January 1, 1979 to the current day. Next, there are two search criteria which are Actors and Events, explained later. And finally, it returns the raw CSV Event records that matched the search criteria.

Google BigQuery

*B*igQuery is a low cost, highly scalable, serverless cloud data warehouse designed to make to make fundamental decisions in a fast way. That is the main reason why all GDELT datasets are available in this platform so it allows you to extract the information needed using simple SQL. It is known for the fast-complex querying, extracting of data and a real-world analysis to be run entirely in the database.

Raw Data files

The third way of taking advantage of this information is downloading all of GDELT to your own computer. Therefore, it allows you to have the entire underlying event and graph datasets in CSV format although deep technical knowledge and extensive experience working with large datasets is required in order to know how to use them correctly.

1.2. CAMEO Ontology

First of all, we need to know the meaning of ontology in order to understand what role it plays here. Formally, it represents knowledge as a hierarchy of concepts within a domain, using a shared vocabulary to denote the types, properties and interrelationships of those concepts. Basically, these are structural frameworks[1] used to organize information.

Applying this concept to this project, CAMEO framework [2], or *Conflict and Mediation Event Observations*, is an event data coding scheme optimized for the study of third-party mediation in international disputes which refers to the analysis of the negotiation facilitated by a neutral mediator between those involved in a conflict in order to reach a reasonable agreement.

1.2.1. Events

CAMEO [2] is specifically designed to code events relevant to the mediation of violent conflict but can also be used for studying other types of international political interactions called **events**.

1.2.1.1. EventID and date attributes

The first few fields of an event record capture are a global unique identifier number, the date the event took place on, and several alternatively formatted versions of this date in order to make it easier to work with them in different analytical software programs which may specify some date format requirements. These attributes are called **GlobalEventID** (integer), **Day** (integer), **MonthYear** (integer), **Year** (integer), **FractionDate** (numeric) [3].

[1] ***Framework***: *platform for developing software applications*

1.2.1.2. Event action attributes

Apart from those ones, there are event action attributes which corresponds to the action of the event, which means what Actor1 did to Actor 2. Moreover, there are attributes that represents several mechanisms for assessing the importance or immediate-term impact of an event. In the table 1.2.1 we can find these fields with a short description of them.

IsRootEvent	Flag[2] that indicates whether the event is below a higher event level or not
EventCode	Raw CAMEO action code described below.
EventBaseCode	Defines the category at level three in the taxonomy.
EventRootCode	Defines the root-level category the code falls under.
QuadClass	It specifies which of the four primary classifications the taxonomy is organized under for the event type.[3]
GoldsteinScale	A numeric score (from -10 to +10) that captures the theoretical potential impact that type of event will have on the stability of a country.
NumMentions	Number of its mentions across all source documents.
NumScores	The total number of information sources containing one or more mentions of this event.
NumArticles	Total number of source documents containing one or more mentions of this event.
AvgTone	Average “tone” of all documents containing one or more mentions of this event.

Table 1.2.1. Event action attributes [3]

[2] **Flag**: variable used as a signal in programming to let the program know that a certain condition has met (true or false)
[3] **Quad classes**: 1=Verbal Cooperation, 2=Material Cooperation, 3=Verbal Conflict, 4=Material Conflict

1.2.1.3. Event geography

The final set of fields encodes the closest reference to each of the two actors and to the action reference. They have the goal of geo-referencing each event along three primary dimensions to the landmark-centroid[4] level. In the table 1.2.2 we can find them with their respective short descriptions.

Actor1Geo_Type	Geographic resolution of the match type
Actor1Geo_Fullname	Full human-readable name of the matched location.
Actor1Geo_CountryCode	2-character code representing the location
Actor1Geo_ADM1Code	2-character country followed by code by the 2-character ADM1[5] code.
Actor1Geo_Lat	Centroid latitude of the landmark for mapping.
Actor1Geo_Long	Centroid longitude of the landmark for mapping.
Actor1Geo_FeatureID	Identifier for the location extracted from GNS/GNIS, a database for locations

Table 1.2.2. Event geography attributes [3]

1.2.1.4. Event codes categories

Each event also captures different aspects of a news article, so the same news article can be referenced in several events with different features highlighted. Therefore, each event is assigned an “EVENTCODE” following the CAMEO framework [2] in order to identify its category that describes the action that Actor1 performed upon Actor2. Below, we can find the different types of events this code describes divided into the main 20 categories and the respective sub-categories.

[4] Two-three-dimensional point taking into account the average position of all points of an object.

[5] A primary administrative division of a country, such as a state in the United States

01. **MAKE PUBLIC STATEMENT**: all public statements expressed verbally or in action which can be specified or not. Example: "U.S military chief General Powell *said* on Wednesday NATO would need to remain strong."

02. **APPEAL**: all requests, proposals, suggestions and appeals. Example: "Indian business leaders Friday *called for* greater impetus towards *free trade* despite mounting tensions between India and Pakistan."

03. **EXPRESS INTENT TO COOPERATE**: offer, promise, agree to, or otherwise indicate willingness or commitment to cooperate. Example: "Syria *says* it *is willing to withdraw* its *troops* from neighboring Lebanon, after fifteen years of effective military occupation."

04. **CONSULT**: all consultations and meetings. Example: "U.S. Secretary of State Warren Christopher *telephoned* Russian Foreign Minister Andrei Kozyrev on Tuesday to discuss efforts to forge a peace settlement in former Yugoslavia, Itar-Tass news agency said."

05. **ENGAGE IN DIPLOMATIC COOPERATION**: initiate, resume, improve or expand diplomatic, non-material cooperation or exchange. Example: "Argentina has *apologized* to Brazil for one of its gunboats intercepting a Brazilian ship in the Beagle Channel, disputed by Argentina and Chile."

06. **ENGAGE IN MATERIAL COOPERATION**: initiate, resume, improve, or expand material cooperation or exchange. Example: "Zambia *extradited* suspected British militant Haroon Rashid Aswad to Britain on Sunday, a senior Zambian government official said."

07. **PROVIDE AID**: all provisions, extension of material aid which can be economic, military, humanitarian, etc. Example: "The United States *continued to send arms* to Pakistan last year, a State Department Spokesman said Wednesday."

08. **YIELD**: all yielding, concessions (to ease, accede, allow, receive, etc.) Example: “The Latvian Constitutional Court *cancelled restrictions* on the use of the Russian *language* on national radio and television.”

09. **INVESTIGATE**: all non-convert investigations such as questioning or inquiring. Example: “Israel’s high court *opened* a landmark *hearing* Wednesday into the legality of secret *interrogation techniques* used against Palestinian detainees.”

10. **DEMAND**: all demands and orders related to cooperation, aids, rights, involvements, etc. Example: “Former Socialist Prime Minister Andreas Papandreou *demanded* immediate *elections* after a special court cleared him of all charges in Greece’s biggest corruption trial this century.”

11. **DISAPPROVE**: express disapprovals, objections, complaints or accusations. Example: “A Saudi businessman *is suing* the United States for damages to his pharmaceutical plant which were caused by a missile attack in August, his American lawyer said.”

12. **REJECT**: all rejections and refusals. Example: “The Turkish government has refused to commit to any direct *assistance* to the US-led *war* against Iraq, citing domestic opposition.”

13. **THREATEN**: all threats, coercive or forceful warnings with serious potential repercussions. Example: “President Yoweri Museveni has *threatened* to *ban* Ugandan *opposition candidates* from participating in the upcoming elections.”

14. **PROTEST:** all civilian demonstrations and other collective actions carried out as protests against the target actor. Example: “Hundreds of thousands of people *blocked streets* in Hong Kong in defiance of Chinese authorities to *demand democratic reforms*.”

15. **EXHIBIT FORCE POSTURE**: all military or police moves that fall short of the actual use of force. Example: "Britain *mobilized army* reservists for a possible war *against* Iraq on Tuesday while UN arms inspectors said they needed more time."

16. **REDUCE RELATIONS**: all reductions in normal, routine, or cooperative relations. Example: "Switzerland said today it had *expelled* two Soviet *diplomats* based in Geneva for spying, adding to a long series of espionage scares."

17. **COERCE**: repression, violence against civilians, or their rights or properties. Example: "Israeli soldiers *arrested* more than 100 Palestinians on Saturday in a security sweep of the Hebron area of the occupied West Bank."

18. **ASSAULT**: use of unconventional forms of violence which do not require high levels of organization or conventional weaponry. Example: "U.S. border patrol agents *sexually abused* illegal Mexican immigrants with impunity, a human rights organization charged on Saturday."

19. **FIGHT**: all uses of conventional force and acts of war typically by organized armed groups. Example: "Palestinian *gunmen attacked* an Israeli village close to the West Bank Sunday and killed an Israeli, public television reported."

20. **USE UNCONVENTIONAL MASS VIOLENCE**: all uses of unconventional force that are meant to cause mass destruction, casualties, and suffering. Example: "Sudan's government *is responsible for mass killings* and other atrocities in the Darfur region, according to a United Nations report."

1.2.2. Actors

We can define an **actor** as one of the two participants in the event. It can be domestic like a country or otherwise, international like an organization, a movement or a company. Therefore, the actor codes are sequences of one or more three-letter abbreviations where each triplet specifies an actor further. For example, one abbreviation can be “IGO” (*Inter-Governmental Organizations*) and a complete sequence can be “IGOUNOHLHWHO” (*Inter-Governmental Organizations / United Nations / Health / World Health Organization*)

1.2.2.1. Features

There are three principles that define the CAMEO actor coding system. The first one is that the codes used are composed of **one or more three-character elements** which are classified into a very extensive number of categories. These ones can be regions, ethnic groups, state actors, etc. [3]

The second principle is that the codes are interpreted **hierarchically**. That means that the codes corresponding to the second element depend on the content of the first element, and the third element depends on the second.

The last concept we have to take into account is that it is based on **standardized codes** and these are not always available. To understand it, we are going to take as an example the use of the United Nations nation-state codes. CAMEO [2] operates with a huge list of these codes so standard codes are generally not available in the sense that they can have acronyms of varying lengths.

1.2.2.2. Actor attributes

There exist some fields that describe attributes and characteristics of the two actors involved in the event. This includes the complete raw CAMEO code for each actor, its proper name and their associated attributes. That means that, for

each actor, there is an array of coded fields indicating geographic, ethnic and religious affiliation and the actor's role in the environment (political elite, military officer, rebel, etc. In the table 1.2.3, there are some examples of these fields. Those fields above are the same for Actor 2.

Actor1Code	The raw CAMEO codes.	*PHL*
Actor1Name	The actual name of the actor.	*MANILA*
Actor1Country Code	The 3-character code for its country affiliation.	*PHL (*Philippines*)*
Actor1Known GroupCode	In case the actor is a known IGO/NGO/rebel organization.	*UNO* (United Nation)
Actor1EthnicCode	Ethnic affiliation	*PER* (Persian)
Actor1Religion 1Code	Religious affiliation	*BUD* (Buddhism)
Actor1Type1Code	The 3-character code of the CAMEO type/role of the actor.	*GOV* (Government)

Table 1.2.3. Actor related attributes [3]

2. TECHNOLOGIES AND APPLIED METHODS

This chapter describes the technical part which includes the methods of data mining, machine learning and complex network analysis. In addition, the software used is defined by Python [4] and MongoDB [5]. Then, all of these technologies are going to be applied in the practical part.

2.1. Data mining

The first method is defined as a process used to extract usable information from a larger set of any raw data. It implies analysing data patterns in large batches of data using one or more software [6]. More specifically, this method is going to be applied in this project for automatic pattern predictions based on trend and behaviour analysis as it is focused on large data sets and databases.

2.1.1. Process

Its process consists of four main steps [7]:

1. **Data collection**: the first thing we must do is to collect some data. As much as information we have the better and easier will be the analysis.
2. **Data cleaning**: it is necessary remove the unwanted making sure that we only have the necessary data since we are getting a large amount of data.
3. **Data analysis**: in this step we need to apply some algorithms in order to analyse and find some patterns.
4. **Interpretation**: finally the analysed data are interpreted in order to explain the results, take the necessary actions or take important conclusions such as predictions.

Figure 2.1.1. Data mining steps

2.2. Machine learning

To understand this new method, it is necessary to understand first the meaning of Artificial Intelligence. AI is a wide-ranging branch of computer science concerned with building smart software modules capable of performing tasks that typically require human intelligence. We can find a lot of recent examples like from chess-playing computers to self-driving cars. There are some main technologies that make it possible and one of them is machine learning [8].

The main purpose of machine learning is the study of teaching a computer program or algorithm how to progressively improve given large amounts of data that can help recognizing some patterns. It is a method of data analysis that refers to the ability of IT systems to independently find solutions to problems [9].

2.2.1. Process

1. **Gathering data**: the first step is to collect all the data that may interest us. The quantity and quality of the data dictate how accurate our model is.
2. **Cleaning data**: then we need to clean, prepare and manipulate the data in order to have only the necessary information.
3. **Model building**: it consists on training the model by choosing the appropriate algorithm and data representation. The clean data are split into two parts: train (to develop the model) and test (used as a reference).
4. **Evaluating the model**: using some metrics to measure objective performance of model.
5. **Data visualization**: transforming results into visual graphs, improving the performance, making a better approximation of how the model will perform in the real world [8].

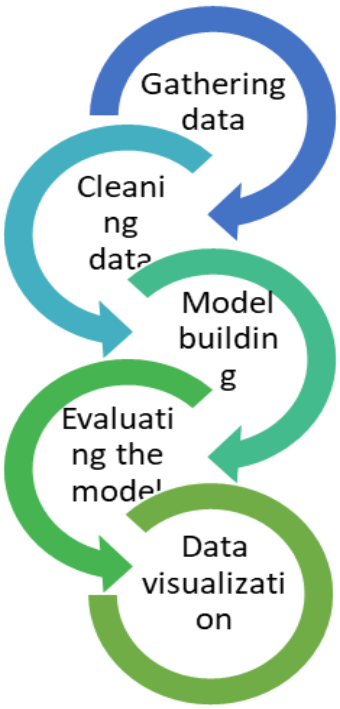


Figure 2.2.1. Machine learning process

2.2.2. Types of machine learning

There are many ways to frame this idea, but there are two major recognized categories, as we can observe in the figure 2.2.2. For each type their most common techniques or algorithm categories are represented, respectively. An explanation of each concept is featured below. [9]

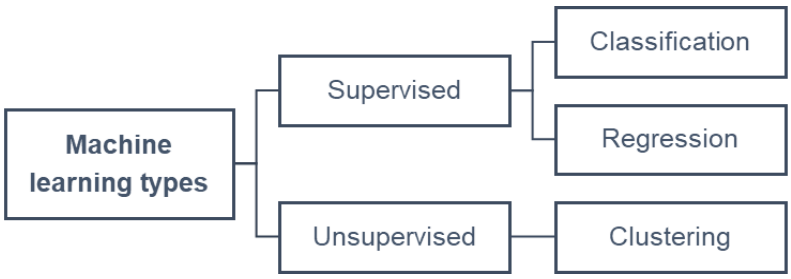


Figure 2.2.2. Machine learning categories and their respective techniques [1]

2.2.2.1. Supervised Learning

This type of learning is the most popular paradigm for machine learning. It builds a model that make predictions based on evidence in the presence of uncertainty, including unknown data. To understand it better, let's analyse the figure 2.2.3. In this case, we have three images as known data and a response telling that these are apples. Next, we have an input which is an unknown data so the purpose of the model is to predict which the new response is.

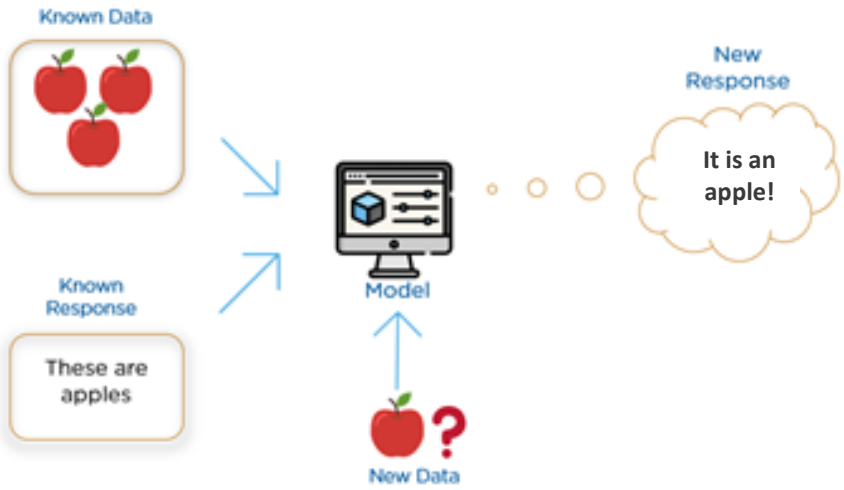


Figure 2.2.3. Supervised learning example [2]

There are two main techniques used for this type of machine learning [10]:

- **Classification techniques** predict discrete responses by classifying the input data into categories. For example, whether and email is genuine or spam, or whether a tumour is cancerous or benign.

- **Regression techniques** predict real or continuous values such as predicting prices of a house given the features of the house such as size, price, etc.

2.2.2.2. Unsupervised learning

Unsupervised learning is very much the opposite of supervised learning. It tries to find hidden patterns or intrinsic structures in data so it is used to draw interferences from datasets consisting of input data without labelled responses.

Let's see the example represented in the figure 2.2.4. In this case, we have several different images as input data although we do not have any kind of specification about them. Therefore, the program will try to find some patterns in order to be able to classify them, for example, by the colour and the silhouette.

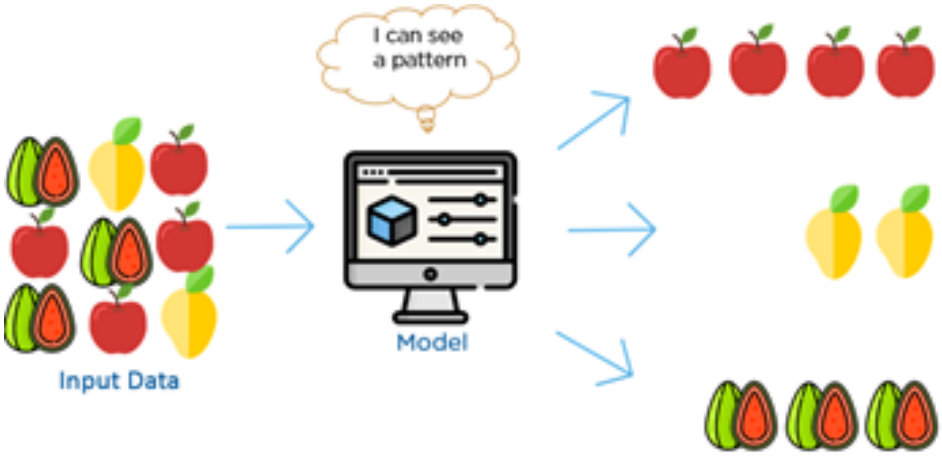


Figure 2.2.4. Unsupervised learning example [2]

Clustering is the most common unsupervised learning technique and its task is to divide the data into groups or clusters such that this data points in the same group are similar to each other. Finally, the top different types of clustering are partitioned and hierarchical algorithms [10].

- Partitioned clustering: used to classify observations, within a data set, into multiple groups based on their similarity. The algorithms require the specification of the number of clusters that need to be generated. In the practical part, K-means clustering is going to be used, where each cluster is represented by the centre of means of the data points belonging to the cluster.

- Hierarchical clustering: it groups similar objects or data into groups or clusters. The endpoint is a set of clusters, were each cluster is distinct from each other. In case of the agglomerative type, initially each data point is considered as an individual cluster. Then, at each iteration, the similar clusters merge with each other until one or K clusters are formed. And divisive type is exactly the opposite, where each data point is separated and considered as an individual cluster. In the end, we will be left with K clusters [11].

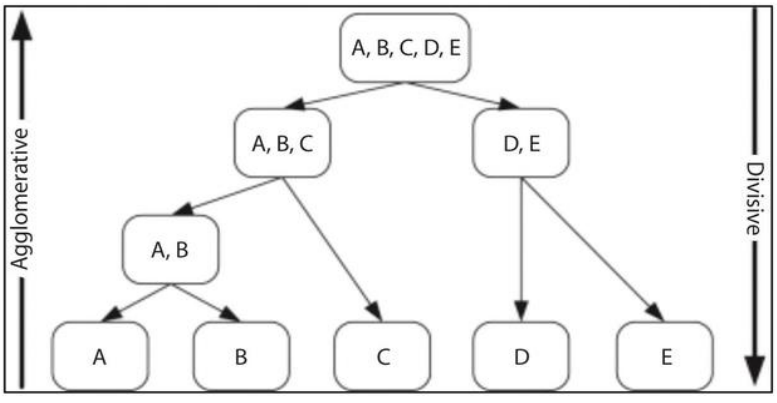


Figure 2.2.5. Example of agglomerative and divisive hierarchical clustering [3]

2.3. Complex network analysis

The complex networks are a set of many connected nodes that interact in different ways with each other. To understand it better let's put the example of social network. In that case, the nodes are people and the connections can be friendship relations. In the same society, we also can define connections in a different way such as two people connected due to the familiar relations (siblings, cousins, etc.). With this example we can realize that we can talk about a complex network as we can define different networks with the same set of nodes depending on how the connections are defined.

Therefore, complex network analysis is a collection of quantitative methods for studying the structure and dynamics of complex networked systems. This analysis is focused on relationships and connections between social entities. It is used in the social and behavioural sciences, as well as in political science, economics, organizational science, etc. [12]

2.4. Tools

As it was commented before, there are several options that we can use with the objective to export this data and analyze it but the one we are going to choose is using a program to combine downloading, importing and preprocessing of the data. The main reason of choosing this option is because it is a great way for exploratory of the analysis of smaller time frames requiring less data. One resource would be the use of *R* [13], a free software environment for statistical computing and graphics. Another option would be the use of Python [4], a general purpose and high-level programming language which can be useful for a lot of implements such as for the development of web applications, network programming, business applications, etc. Nowadays, it has become a popular language for rapidly working with large datasets as it provides a lot of tools and facilities for data science, data mining and machine leaning at a large-scale.

In conclusion, we are able to use both of these software environments although we are going to use the second option. After doing some research about them, Python [4] seems to have more options aside from data statistics also, it gained popularity for its code readability, speed and many functionalities. Moreover, from a subjective point of view, it is going to be easier as I have already been in touch with this language for machine learning.

2.4.1. Python

Python [4] is a general-purpose, versatile and modern programming language which is quite concise and easy to read. Python libraries are very useful to create systems based on machine learning.

Python libraries

Python [4] relies on modules and libraries which are tools that allow it to be used more easily for machine learning. Some of those are *Anaconda, Pandas, NumPy* and *Scikit Learn.* For a better understanding, these are described below.

- ***"Anaconda"*** [14] is the standard, free and open-source platform for Python [4] specialized for scientific computing as it allows accessing and managing data science and machine learning libraries and packages. In this project, we are going to use more specifically the free integrated development environment (IDE) called ***Spyder***, which is found in the Anaconda distribution.

- ***"Pandas"*** [15] is a flexible open source data analysis and manipulation tool. It is designed for the analysis of structured data which means any data that reside in a fixed field within a record or file. Pandas is very useful for data contained in databases, spread sheets such as Excel sheets, or another type of statistical datasets.

- ***"NumPy"*** [16] is Python package that is very useful for the development of mathematical operations such as linear algebra, Fourier transform and random number capabilities.

- ***"Scikit Learn"*** [17] is a free software machine learning library for Python [4] that features several algorithms explained before, such as classification or clustering. It is designed to interoperate with the Python numerical and scientific libraries like NumPy [16].

2.4.2. MongoDB

MongoDB [5] is a distributed database built for modern application developers and for the cloud era. This program uses JSON-like documents with *schema*, which is a language-independent data format as shown in the figure 2.4.1. Imagine that we have a database that contains information about people. MongoDB [5] will show each person information like in the figure, with the respective attributes for each one.

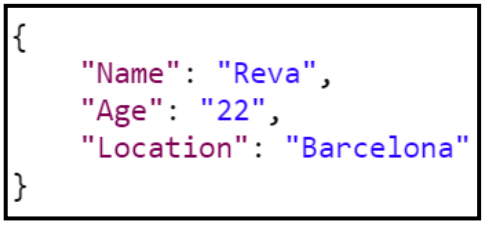


Figure 2.4.1. Example of JSON data format

The main advantage of MongoDB [5] is that it is classified as a NoSQL database program, which means that is designed to provide flexible *schemas* for the storage and retrieval of data beyond the traditional table structures found in relational databases.

In the practical part of this project, we are going to explain how it works in a more specifically way as we are going to use it to storage of our database.

3. MAIN IDEA

Once we provided more knowledge, it is time to explain how we are going to apply the technologies mentioned before and why it is so important to know the theory explained.

To achieve it, the development of this study starts with how the data work and what we should do in order to be able to use it. In addition, the system architecture is defined in order to make it easier to understand how the dataset works. Some examples for that would be the number of events grouped by their kind, the countries that are mentioned the most, the correlation between these countries or the most common types of events.

Then, we will start digging into the very special topic selected by achieving more specific measures, their analysis and also, the evolution in time which can be in days, weeks or months. A measurement would be, for example, the correlation between Actor1 and Actor2 taking into account a specific kind of event or country. Moreover, another interesting analysis can be how the virus affected to other type of events. For instance, being able to discover if this pandemic helped to minimize the political conflicts, the worldwide pollution problems, etc. Or, it may have had negative affectations in other sectors like economy or death rate.

Finally, with the values of these measures we are going to create vectors which will be clustered taking into account different. Consequently, we can also think of patterns that can be made through the analysis got from these measures.

Moreover, there will be static and dynamic clusters as it is going to be represented globally and month by month, respectively. Apart from that, there are going to be some different dimensions for them and a comparison with two different algorithms mentioned before: K-means and hierarchical clustering.

The next step would be the clustering for ten specific countries that has been chosen considering the importance of this country worldwide, the correlation between them and how they have been affected by the Coronavirus. Therefore, they are going to be analyzed in order to see how they evolve taking into account external aspects apart from the epidemic.

Another fact that can be analyzed is the correlation between Actor 1 and Actor 2. What makes it interesting is the information that can be provided if we set a specific country as the Actor 1 and calculate its percentage of events. Then, repeat it again as the Actor 2 and finally observe the difference between them. Or another simple analysis would be calculating the five most common codes for both actors.

Finally, the same clustering is going to be repeated but considering all countries instead of just ten so that we can appreciate how stable are them and how the strength between these countries has been affected considering the spread of COVID-19.

4. DEVELOPMENT

After understanding the main idea, let us start describing the development. This chapter explains how the data work and what we should do in order to be able to use it. In addition, the system architecture is defined.

4.1. How to use the data

4.1.1. Data format

The first step we need to do in order to analyze an event is to decide which method will be used in order to extract the information needed. As it has been explained before, there are three possibilities but, for the moment, we are going to use raw data files. That means that we can directly download the zipped CSV files to build a dataset or database from them although there is another option to download these files which is using some software environment.
Next, we need to consider the time of the information we want to gather as we are going to use the GDELT data event files divided per day. That means that we are able to export a CSV file with the information gathered day by day.

Taking into account that the first case of this virus was proclaimed on 7th January, the date range will be from the 1st of January until 31st June. Therefore, we will be able to analyze its impact during the first three months.

Field Name	Column ID
GLOBALEVENTID	0
SQLDATE	1
MonthYear	2
Year	3
FractionDate	4
Actor1Code	5
Actor1Name	6
Actor1CountryCode	7
Actor1KnownGroupCode	8
Actor1EthnicCode	9

Figure 4.1.1. File with field names

Unfortunately, these files do not contain the fields' names which are very important information in order to know their meaning. For that, we will have to use another XLSX[6] file that contains the name of all the attributes with their respective column identifier, just like in the *figure* 4.1.1. We can only appreciate the

[6] XLXS is a file extension for the documents created with Microsoft Excel, since 2007.

first ten fields although there are 58 in total. Once we have this information, we are able to export the database. For the moment, we are just going to show the database for one random day. Consequently, we obtained something like the figure 4.1.2.

GLOBALEVENTID	SQLDATE	MonthYear	Year	FractionDate	Actor1Code	Actor1Name
895862728	20100102	201001	2010	20.100.055	CRM	CRIMINAL
895862729	20100102	201001	2010	20.100.055	CRM	CRIMINAL
895862730	20100102	201001	2010	20.100.055	CRM	CRIMINAL
895862731	20100102	201001	2010	20.100.055	ISR	ISRAEL
895862732	20100102	201001	2010	20.100.055	ISR	JERUSALEM
895862733	20100102	201001	2010	20.100.055	ISRMED	JERUSALEM POST
895862734	20100102	201001	2010	20.100.055	UAF	GUNMAN
895862735	20100102	201001	2010	20.100.055	UAF	GUNMAN
895862736	20181231	201812	2018	20.189.890		
895862737	20181231	201812	2018	20.189.890		
895862738	20181231	201812	2018	20.189.890	BUS	PRODUCER
895862739	20181231	201812	2018	20.189.890	BUS	PRODUCER
895862740	20181231	201812	2018	20.189.890	BUS	COMPANY
895862741	20181231	201812	2018	20.189.890	BUS	INDUSTRY
895862742	20181231	201812	2018	20.189.890	BUS	EMPLOYER

Figure 4.1.2. CSV file with the events

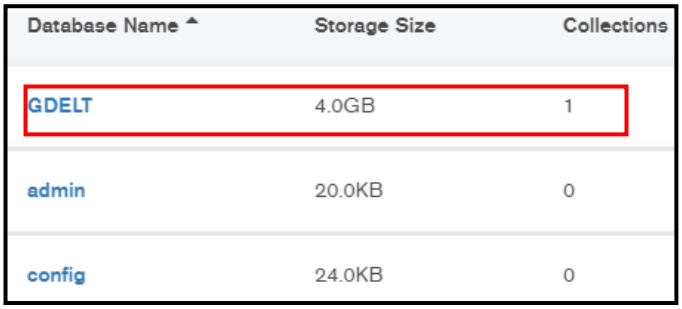
As we can appreciate, we find a type of table where each row represents the different events. And consequently, each column represents a code for them. It is necessary to remark that the original file does not contain the first row with the field names, as mentioned before. Also, just to show the data, we needed to change its format as all the information is presented separated with semicolons, which makes it almost unreadable. For the whole description of all fields, you can have a look at the GDELT Event Codebook where each event's encoding is very detailed. So, once we understand how this file is composed, it is the time to operate with it.

4.1.2. Downloading and extracting

The first thing that we have to do is to download all the dataset from the website. For that, a script was created using Python [4] and some of its libraries where we needed to specify the date range [10]. After downloading it, we have 91 compressed files which we need to unzip in order to use the CSV files. Therefore, once it is unzipped, we are able to have a look to all the files which will be stored in another folder temporally.

4.1.3. Importing the data to MongoDB

In order to import the data to MongoDB [5], first we need to write a script to create a connection with its server. Then, we need to define a database and at least, one collection inside it. Also, we must specify the local path of the data and then insert all into this new database recently created. As result, we will have a database created with the name of “GDELT” and containing only one collection, as we can observe in the figure 4.1.3. As an interesting fact, we can see that the storage size is really huge and that is one of the main reasons why we are using this database program.



Database Name	Storage Size	Collections
GDELT	4.0GB	1
admin	20.0KB	0
config	24.0KB	0

Figure 4.1.3. The database imported to MongoDB

Consequently, we obtain a database with all the events as showed in the following figure. The first column represents the identifier for each event which is totally unique; each row represents a different event and the rest of the columns contain the events attributes. However, in the figure 4.1.4 we can only appreciate the three first attributes when there are 58 columns in total.

	_id ObjectId	SQLDATE String	MonthYear String	Year String
1	5e970fa6447752486cc095d4	"20190101"	"201901"	"2019"
2	5e970fa6447752486cc095d5	"20190101"	"201901"	"2019"
3	5e970fa6447752486cc095d6	"20190101"	"201901"	"2019"
4	5e970fa6447752486cc095d7	"20190101"	"201901"	"2019"
5	5e970fa6447752486cc095d8	"20190101"	"201901"	"2019"
6	5e970fa6447752486cc095d9	"20190101"	"201901"	"2019"
7	5e970fa6447752486cc095da	"20190101"	"201901"	"2019"
8	5e970fa6447752486cc095db	"20190101"	"201901"	"2019"
9	5e970fa6447752486cc095dc	"20190101"	"201901"	"2019"

Figure 4.1.4. Data imported to MongoDB

4.1.4. Cleaning the data

Once we have the entire database, it is necessary to clean it in order to achieve datasets with only useful information. Therefore, we must take into account what is the analysis about and which fields are really important for us and for that, we need to code a script indicating the unwanted documents and the fields we want to delete. For example, there is a huge quantity of documents that are related to past events, older than 2019, which are not of our interest. Therefore, all the events that are not related to December, 2019 or 2020 have been removed.

Once we apply it, we can appreciate a quite important difference especially on the database size, as it is reflected on the figure 4.1.5.

Collection Name	Documents	Avg. Document Size	Total Document Size
Events	24,837,421	1.6 KB	37.5 GB
Collection Name	**Documents**	**Avg. Document Size**	**Total Document Size**
Events	24,073,802	850.3 B	19.1 GB

Figure 4.1.5. Before and after cleaning the database

4.2. System architecture

The following system architecture defines the structure and the behavior of the model designed for this project in order to accomplish the practical part. Observing the figure 4.2.1, the first step is to extract the data from the global database of society, GDELT. For that, it is necessary to create a Python script through the environment Spyder that is going to export the data into CSV files.

These data have to be sent to a database so we can use it for our analysis. To accomplish it we have to repeat the same process as the previous one, creating a Python script to import the data to MongoDB [5].

Once we have the data prepared, we are ready to make the required analysis by programming different Python scripts with the Spyder environment which lets us also visualize all the necessary graphs.

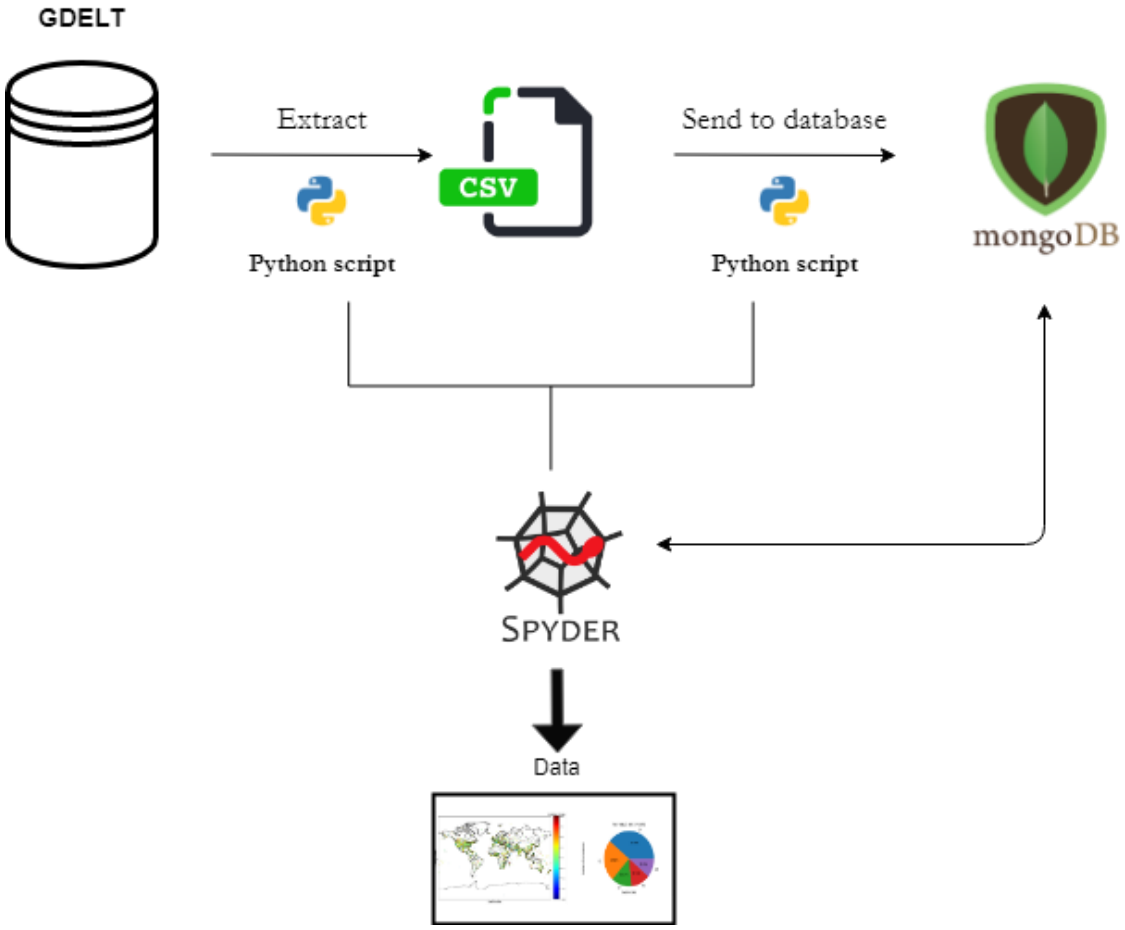


Figure 4.2.1. System architecture of the practical part of the project

5. GENERAL ANALYSIS

We are going to start with simple analysis of the data in order to get in touch with the information gathered and to create some general context. Therefore, the first issue we need to make clear is the date range of the events. As it was mentioned before, we selected the data from January 2020 until June 2020, included. However, it does not mean that this data do not contain events from another year, decade or century.

5.1. Total number of events

The first thing we need to calculate is the total number of events that we have in these three months. Therefore, compiling the corresponding script we found out that there are 24,073,802 events, which is a very huge quantity of events to take into account.

```
In [1]: runfile('C:/Users/Reva/Desktop/ANALYSIS_01/04_basic_commands.py',
wdir='C:/Users/Reva/Desktop/ANALYSIS_01')
24073802
```

Figure 5.1.1. Result of number of total events

Moreover, we can appreciate that we have more or less the same quantity of events per each month, as shown in the figure 5.1.1. However, there is a big difference between how the events are increasing from January to March and then, all of a sudden, the number decreases a lot for the month of April and May.

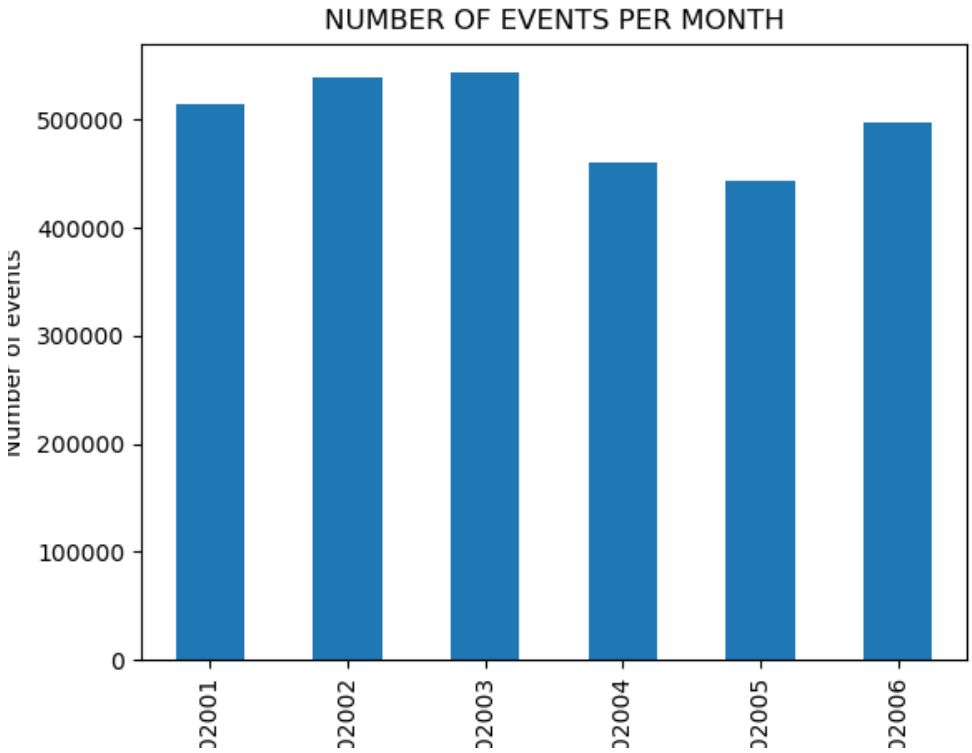


Figure 5.1.2. Number of events per month

5.2. Kind of events

In this section we are going to analyse the kind of events explained in the theoretical part. The first thing we need to see is the quantity of events for each type and for each month available, as represented in the figure 5.2.1.

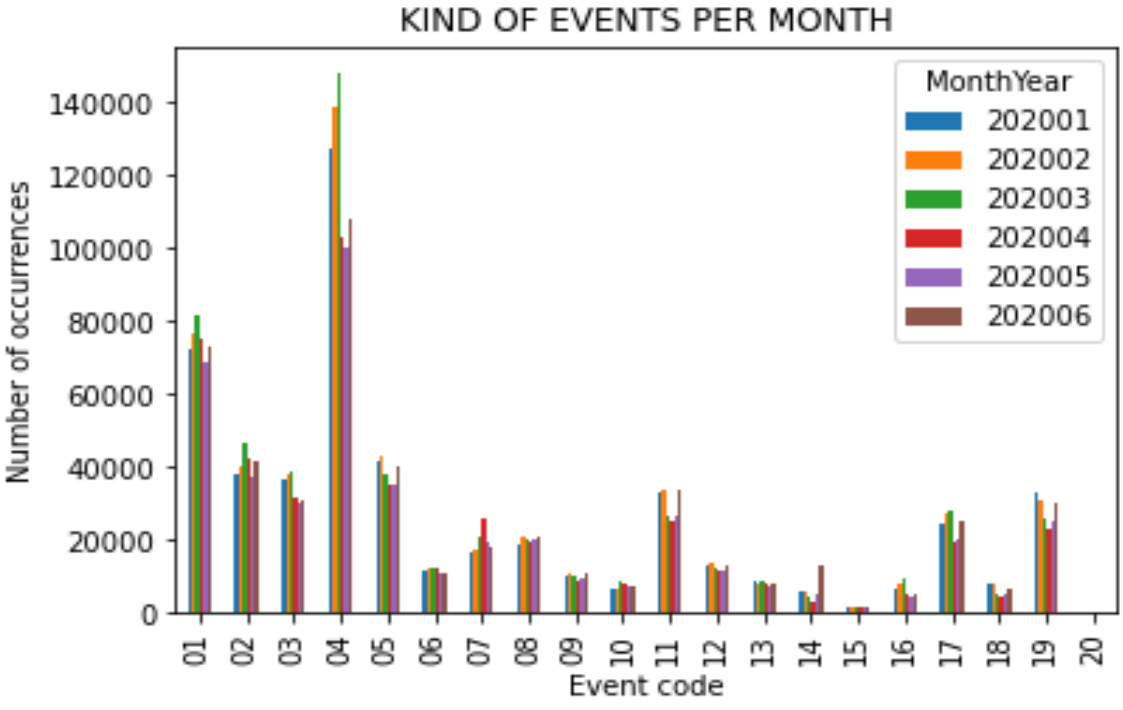


Figure 5.2.1. Quantity of events for each type and month

There is a clear type of event that stands out from the rest, which is the 04. It represents the documents related to consultations and meetings, which is very understandable as it encloses the most simple and common events such as a discuss by telephone, making a visit, engaging in mediation or negotiation, etc.

Moreover, if we look into the differences between each month, approximately the half of them increased the number of events from January to April. These are making a public statement (01), making an appeal or request (02), expressing intent to cooperate (03), consulting (04), providing aid (07), demand (10), reducing relations (16) and coercing (17). After that, the ones mentioned previously decreased considerably except for one which has his highest mark for the month of May. This one is providing aid (07), which makes a lot of sense if we consider the pandemic. Some others decreased their quantity such as rejecting (12), protesting (14) or assaulting (18) although they increased again in June. We can say that this month was the best one after a long time as the pandemic was more controlled in most countries. Therefore, this type of evens started taking place again.

And finally, we have some of them that barely appear. One example would be exhibit military or police power (15) which includes increasing police alert status, mobilize or increase armed forces, etc.

In the next figure 5.2.2 we have the top kind of events with the most percentage of data for the whole three months.

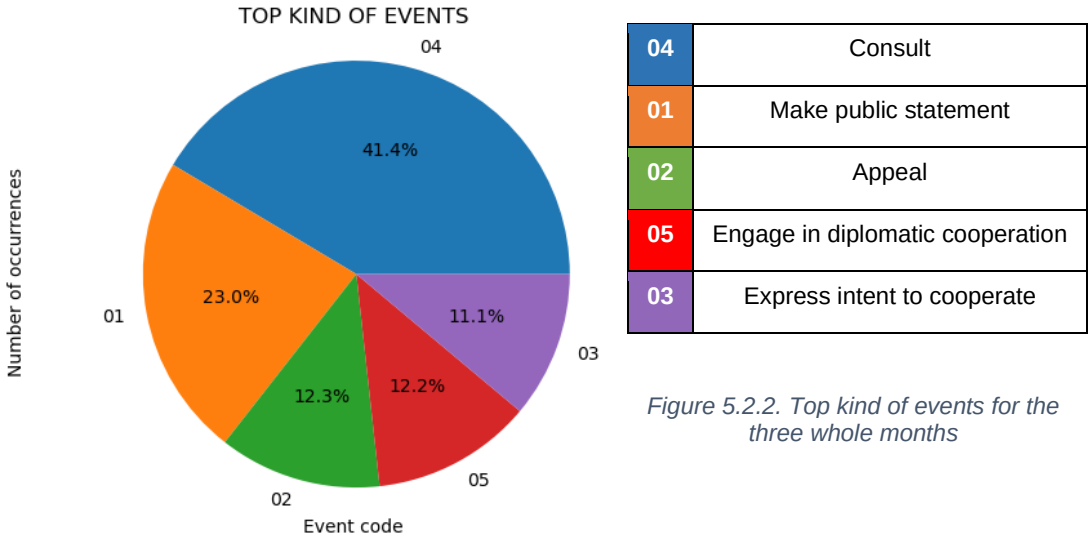


Figure 5.2.2. Top kind of events for the three whole months

The most common one is the consulting type, as we realized before. The second most common is making a public statement (01) which includes declining to comment on a situation, make pessimistic or optimistic comments, considering policy options or expressing some accord or agreement. Next, with almost the same percentage of data, we find making an appeal or request (02) and engaging in diplomatic cooperation (05). Taking into account the previous figure 5.2.2, notice that the first one (02) increased the quantity from January to March, although the other one (05) decreased quite a bit from February to March. And finally, the fifth most common one is expressing intent to cooperate (03).

Next, there is represented in the world map the same kind of events shown in the previous figures. We can appreciate that the most common kind of events are the first five ones, as we could see in the figure 5.2.2 and that there are some place which we barely have information about. These are the whole part of South America, the north of the continent Africa and some countries such as Canada and Russia.

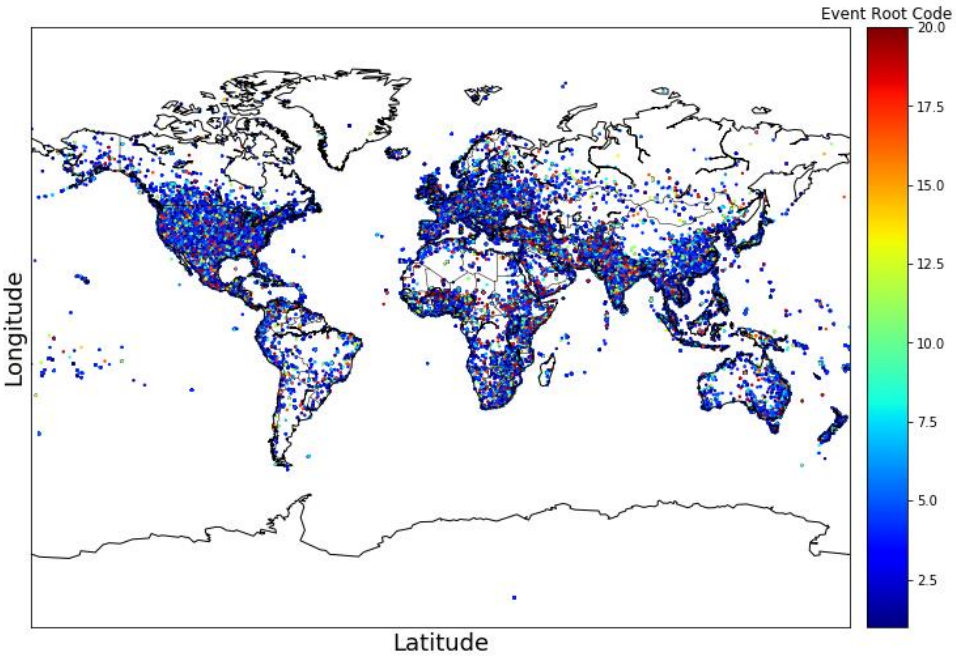


Figure 5.2.3. World map with the event root codes

5.3. Analysis related to China

In the next figure, there is represented the evolution of the number of events related to China. More specifically, it shows the events that have China as the code for Actor 1, which means that in these events this is the country causing the action.

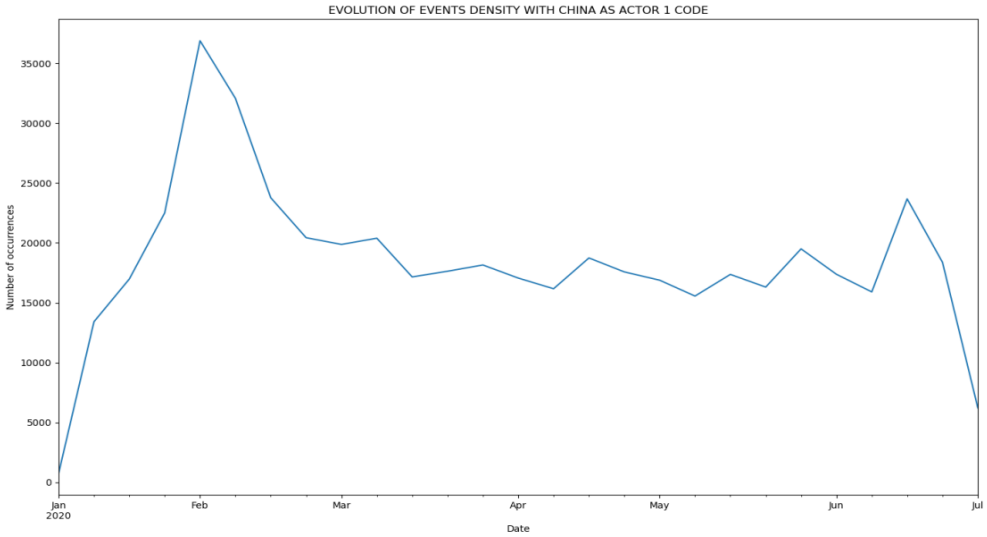


Figure 5.3.1. Evolution of events with China as Actor 1

There is a clear peak at the beginning of February with the highest quantity of data. We can suppose that at that time is when the country had an important impact related to the pandemic. After that, it has been decreasing until the month of June where there is represented another peak, lower than the first one. However, to understand better the impact that it had respect to the other countries, let us analyze the figure 7.5.2. There is represented the correlation between China and the other countries with most events. Also, the evolution per months can also be appreciated.

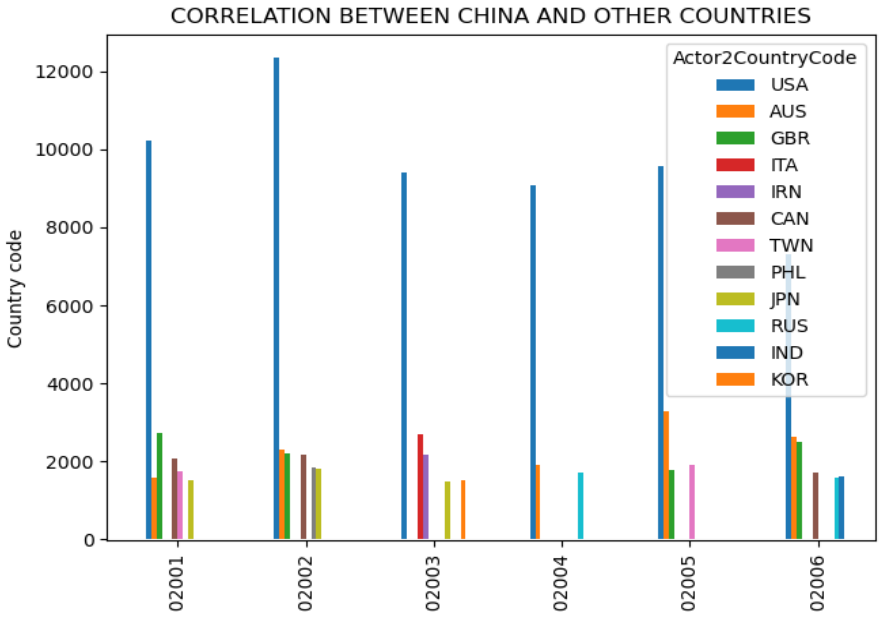


Figure 5.3.2. Correlation between China and other countries

Having a general look, there is a lot of a variety within the countries and their evolution in time. On January the countries with more presence are USA, Australia, United Kingdom, Iran, Taiwan and Japan. During the second month they stayed more or less the same except for some changes like the increase of USA and Australia and the new appearance of Philippines. The big change comes when we talk about the next two months where there is a huge increase for the country of Italy, the country where Coronavirus affected the most after China. On April, Russia appeared due to the fact that Russia's new coronavirus infections have risen quickly in April according to the article[7] written by Tom Balmforth. And finally, during the months of May and June, Australia cases where China is the one causing the action increased quite a lot along with Canada and India.

[7] https://es.reuters.com/article/worldNews/idUSKBN2221IF

6. IDENTIFICATION OF PATTERNS AND GROUPS OF EVENTS

Now that we can understand better the data after the general analysis, let us introduce the clustering, the most common technique to use unsupervised learning. It is used for exploratory data analysis to find meaningful structures, hidden patterns, or groups in the dataset. Therefore, different algorithms are going to be used in order to compare them and conclude which is the most accurate one.

With this method of machine learning, it is going to be tried to figure out if the clusters change as the time goes by considering the spread of COVID-19, so it will be represented static and dynamic clusters. Therefore, we will be able to know which data have been more affected taking into account that there are going to be used three principle measures.
Given a vector of two dimensions, the measures "GoldsteinScale" and "EventRootCode" are going to be used in order to classify and cluster the data depending on the 20 different types of events previously mentioned and the Goldstein scale from -10 to 10. Then, the same process is going to be repeated changing the measure of Goldstein scale for average tone to see if there is a better correlation with the type of event or not. And the last combination will be the Goldstein scale and the average tone together to find out some kind of correlation as well.

All of these clusters are going to be calculated for each month in order to analyze the evolution of the data and their correlation with the epidemic.

Finally, given a vector of three dimensions with a length of 12,784,449 events, the measures "GoldsteinScale", "AvgTone", "EventRootCode" are going to be used in order to classify and cluster the data depending on the 20 different types of events, the Goldstein Scale from -10 to 10 and the Average Tone with the same range as the previous measure.

The last cluster is going to be repeated with the other algorithm in order to compare both of them and be able to decide which one is more accurate.

6.1. K-means

The first algorithm for clustering is K-means. This process begins with k centroids initialized randomly where they are used to assign points to its nearest cluster. In order to obtain the most accurate clusters, it is necessary to calculate the most optimal number of clusters (*k*).

To do that, there are several methods that help you get this number. One of them is called elbow method and it helps to choose the optimum value of number of clusters (*k*) by fitting the model with a range of values for 'k'. In our case, the range selected for this value is from 1 to 10, as it is represented by the axis x. Consequently, for each value of k it computes an average score for all clusters. By default, the score is computed as the sum of square distances from each point to its assigned center [18].

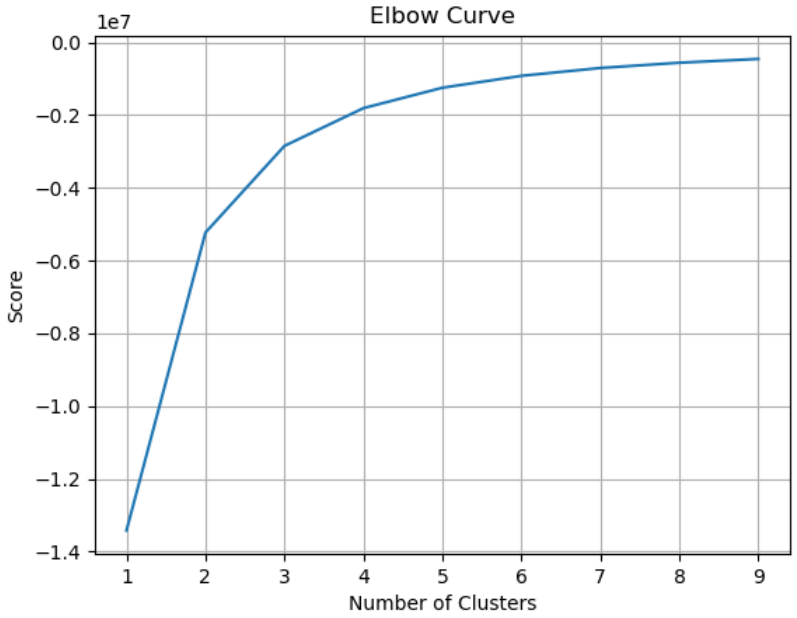


Figure 6.1.1. Elbow curve method to determine the optimal number of clusters

Observing the results obtained on the figure 6.1.1, we found out that the optimal number of clusters at the "elbow" is 3, which is the point after which the inertia starts decreasing in a linear fashion.

The first clustering is calculated between the average tone and Goldstein scale. The figure 6.1.2 shows the clustering considering all data from January to June.

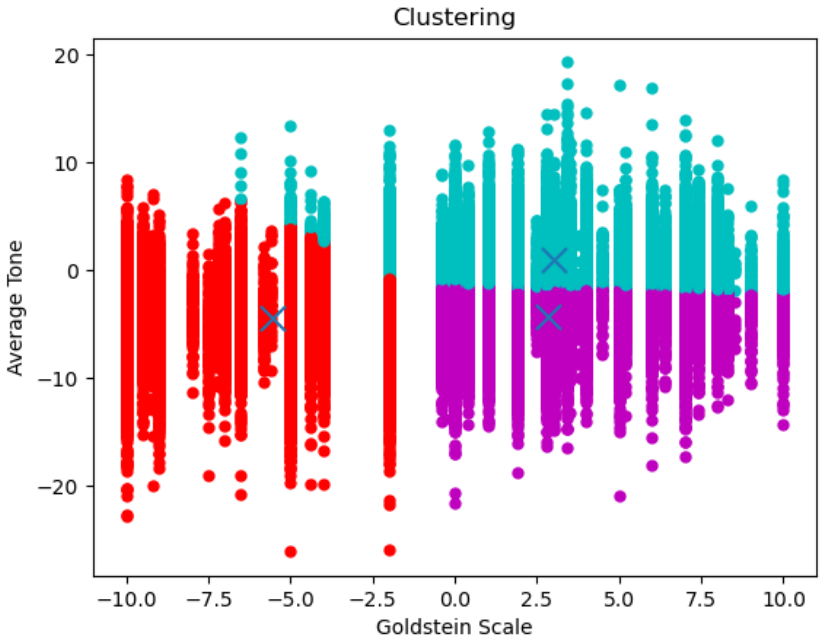


Figure 6.1.2. Clustering of all dataset for Goldstein Scale and Average Tone

To have a better knowledge about the evolution of how the data are clustered considering these two measures, there is represented the same graph calculated for each of the six months.

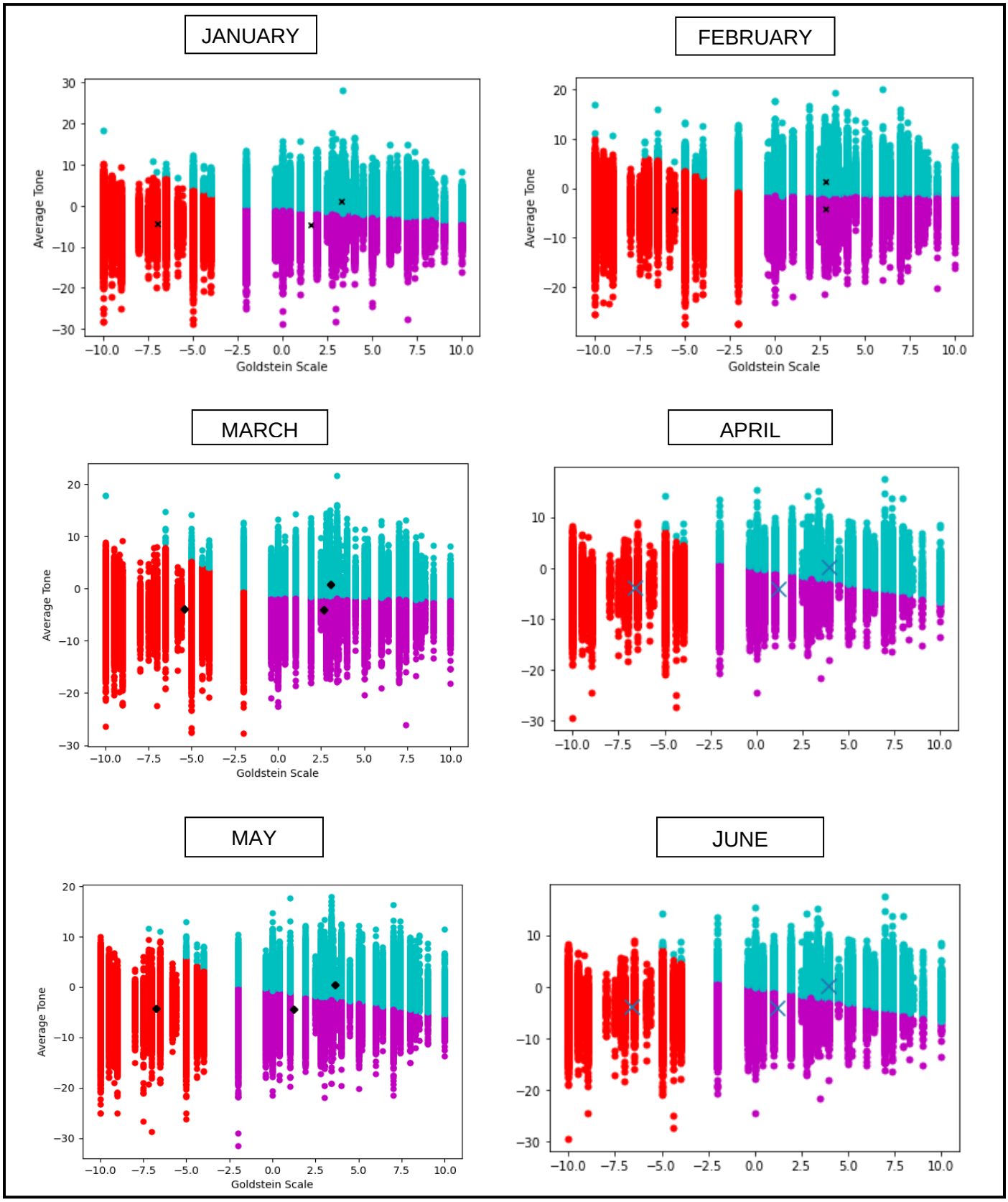


Table 6.1.1. Clustering of all dataset for the measures Goldstein scale and average tone for each of the six months

The first thing to note is that they have been clustered in 3 groups. Despite that the clusters are quite similar in all these months, there are small differences between them. To analyze it, it is necessary to consider the table 6.1.2 which gives us a lot of information about the graphs.

The first measure is the average value for Goldstein scale. The values are really neutral, although it seems like it increased from January to April and then, it starts to decrease again, having the worst value for the month of June. Consequently, the same happens with the average tone. Now, relating these values with the pandemic, the best scores were achieved at the same moment when there was happening a lockdown in most of the places.

Next, let us analyze the clusters one by one. In order to interpret the maximum and minimum values, it is necessary to know that they have been calculated as the sum of both coordinates and then, taking the maximum and minimum values, respectively. Considering the first ones, we can appreciate for each cluster how the numbers increase mainly for the month of April and May, having the best peak for the month of April. In the case of the minimum values, they decrease as the time goes by, having the peaks on May and June.

The following values are the centroids and we can observe how we have the same behavior as the average values. The best values for the centroids are achieved during the months of April and May, which makes us think what the main reason can be.

Finally, there is represented the standard deviation for each cluster. It measures the amount of variation or dispersion of a set of values. A low standard deviation means that they tend to be close to the mean of the set, while a high standard deviation indicates that the values are spread out over a wider range. Therefore, we get to know how accurate the clusters are.

MONTHS	K	Nº OF EVENTS	MEAN VALUE		For each cluster							
					MIN VALUES		MAX VALUES		CENTROIDS		STANDARD DEVIATION	
			GS	AT	GS	AT	GS	AT	GS	AT	GS	AT
January	3	4.07M	0.48	-2.3	-9	-28.74	-6.5	6.85	-6.93	-4.52	2.45	3.49
					-2	-20.83	10	-4.15	1.60	-4.63	2.61	2.27
					-2	-0.44	7	13.72	3.25	1.04	2.68	2.34
February	3	4.24M	0.59	-2.12	-5	-26.07	-9.5	9.22	-5.57	-4.49	2.53	2.12
					-0.4	-19.23	10	-1.76	2.81	-4.29	2.94	3.43
					-2	-0.60	3.4	18.01	2.89	1.32	2.39	2.24
March	3	4.32M	0.84	-2.19	-10	-21.97	-7	7.84	-5.45	-4.05	2.29	2.06
					0	-20	10	-2.35	2.71	-4.12	2.5	1.99
					-2	-0.08	3.4	21.43	3.01	0.74	2.79	3.38
April	3	3.60M	0.97	-2.19	-10	-29.41	-6.5	8.87	-6.63	-3.94	2.32	2.41
					0	-24.39	10	-6.67	1.23	-4.07	2.43	3.52
					-2	1.09	7	17.44	3.92	0.27	2.44	2.21
May	3	3.38M	0.69	-2.23	-6.5	-38.46	-6.5	8.81	-6.76	-4.21	2.52	2.20
					-2	-21.73	10	-5.59	1.29	-4.36	2.42	3.59
					-2	0.19	7.4	14.28	3.61	0.48	2.49	2.37
June	3	3.94M	0.37	-2.56	-9	-35.46	-10	11.31	-6.87	-4.46	2.52	2.36
					-2	-21.63	10	-5.67	1.21	-4.67	2.34	3.44
					-2	-0.13	7.4	16.67	3.53	0.38	2.56	2.27
ALL DATASET	3	24M	0.65	-2.26	-10	-22.72	-6.5	6.46	-5.56	-4.35	2.93	3.33
					0	-21.62	10	-1.87	2.82	-4.22	2.41	2.13
					-2	-0.53	6	16.89	2.30	0.99	2.55	2.02

Table 6.1.2. Table with the measures of the clusters from January to June for Goldstein scale and average tone

The next cluster is between Goldstein scale and event root code. In this case we can appreciate that the clusters are more separated than previous one. Analyzing the first cluster represented with purple color, it can be said that the first half of types of events have the best scores regarding the Goldstein scale. Therefore, these measure decreases as the type of events number increases.

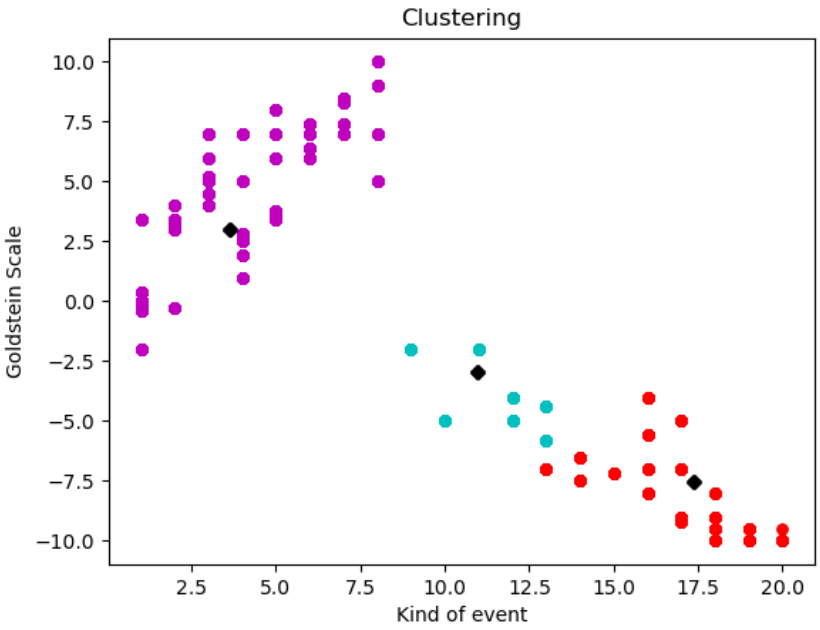


Figure 6.1.3. Clustering between Goldstein scale and event root code

Next, let us analyze the same clustering for every of the six months from January to June as it was done before with the previous clustering.

They have been clustered in 3 groups, considering the optimal number of clusters calculated previously. Despite that the clusters are quite similar in all these months, there are small differences between them. To analyze it, it is necessary to consider the table 6.1.3 which gives us a lot of information about the graphs.

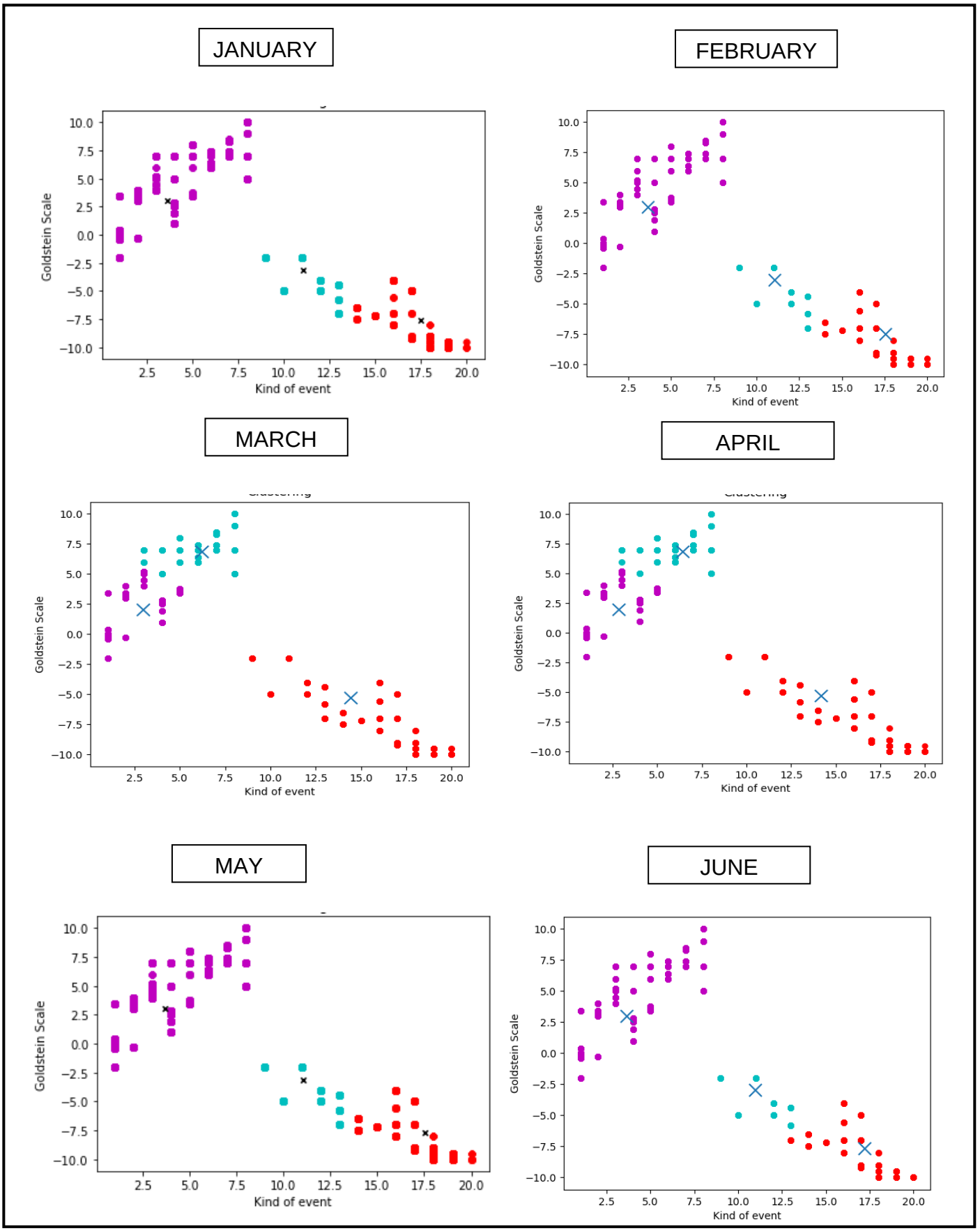


Table 6.1.3. Clustering of all dataset for the measures Goldstein scale and kind of event for each of the six months

The first measure is the average value for kind of event. Considering that there are 20 different types of events, the fact that the average value for all months goes from 6.29 to 6.96, according to the table 6.1.4 it means that there are more events for the first half than the second one. The behavior of the data considering the Goldstein scale is similar than the previous measurements; it increases for the months of March and April.

Next, let us analyze the clusters one by one. Considering maximum and minimum values, let us say that there is not a remarkable difference between one month and another, as it remains almost the same.
The following values are the centroids and we can observe how we have the same behavior as the average values. The best values for the centroids are achieved during the months of April and May, which makes us think what the main reason can be.

Finally, there is represented the standard deviation for each cluster. As it was mentioned before, it measures the amount of variation or dispersion of a set of values. Therefore, we get to know how accurate the clusters are.

GS – Event Code	K	Nº OF EVENTS	MEAN VALUE		For each cluster							
					MIN VALUES		MAX VALUES		CENTROIDS		STANDARD DEVIATION	
			EC	GS	EC	GS	EC	GS	EC	GS	EC	GS
January	3	4.07M	6.56	0.7	1	-2	8	10	3.62	3.0	1.97	2.4
					10	-5	11	-2	11.03	-3.1	1.16	1.37
					14	-7.5	17	-5	17.5	-7.55	1.52	2.35
February	3	4.24M	6.70	0.62	1	-2	8	10	3.63	2.98	1.91	2.36
					10	-5	11	-2	11.03	-3.03	1.14	1.36
					14	-7.5	17	-5	17.53	-7.48	1.45	2.38
March	3	4.32M	6.35	0.83	1	-2	5	3.8	2.96	2.02	1.39	1.46
					4	5	8	10	6.20	6.86	1.55	0.96
					10	-5	17	-5	14.41	-5.29	3.46	2.81
April	3	3.60M	6.29	0.97	1	-2	5	3.8	2.85	1.99	1.46	1.52
					3	6	8	10	6.40	6.84	1.45	0.94
					10	-5	17	-5	14.18	-5.26	3.53	2.91
May	3	3.38M	6.56	0.72	1	-2	8	10	3.64	3.04	2.03	2.43
					10	-5	11	-2	10.99	-3.10	1.18	1.36
					14	-7.5	17	-5	17.53	-7.65	1.53	2.32
June	3	3.94M	6.96	0.36	1	-2	8	10	3.62	3.00	2	2.43
					10	-5	11	-2	10.96	-2.96	1.11	1.26
					13	-7	17	-5	17.18	-7.63	1.83	2.20
ALL DATA SET	3	24M	6.62	0.67	1	-2	8	10	3.62	3.02	1.97	2.41
					10	-5	11	-2	10.99	-3.08	1.17	1.38
					14	-7.5	17	-5	17.48	-7.54	1.54	2.34

Table 6.1.4. Table with the measures of the clusters from January to June for Goldstein scale and kind of event

The next cluster is between the average tone and event root code. In this case we can appreciate that the clusters are more similar to the clustering between average tone and Goldstein scale. From a general view, there is not a clear separation between any of the clusters. Despite of that, the first half of type of events is divided into two groups where the only difference is that one of them has positive values for average tone and the other one negatives. Then, the third cluster represents all the second half of types of events.

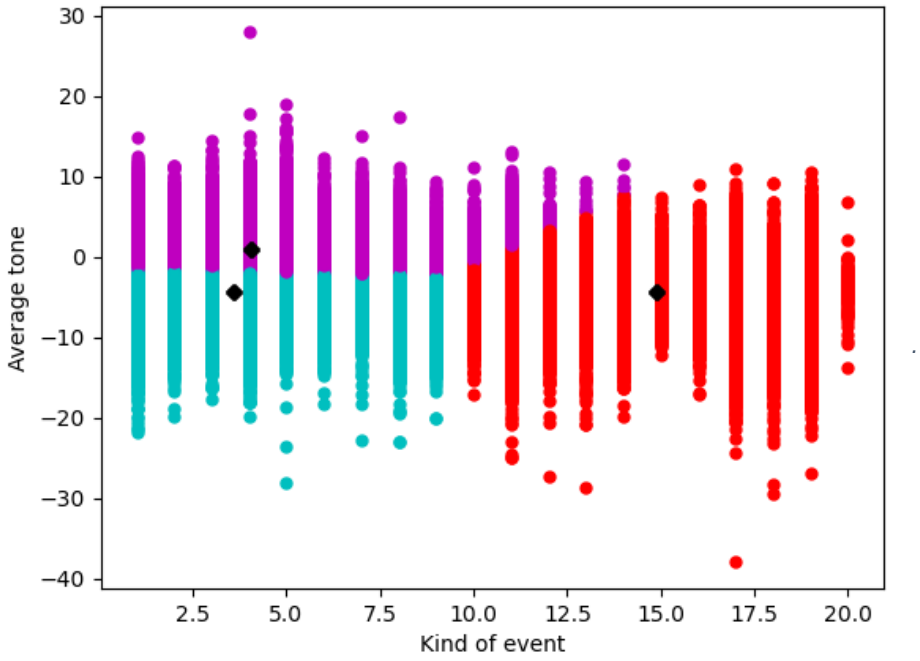
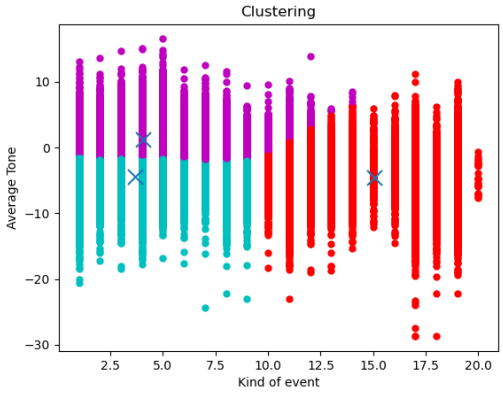


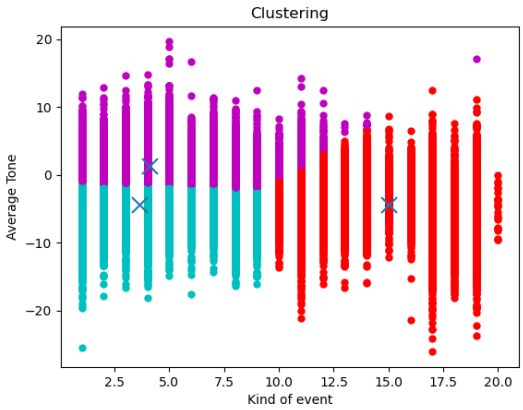
Figure 6.1.4. Clustering between average tone and event root code

Next, let us analyze the same clustering for every of the six months from January to June as it was done before with the previous clustering.

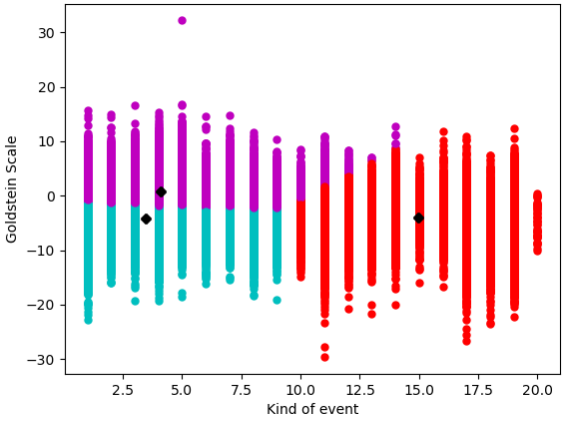
JANUARY



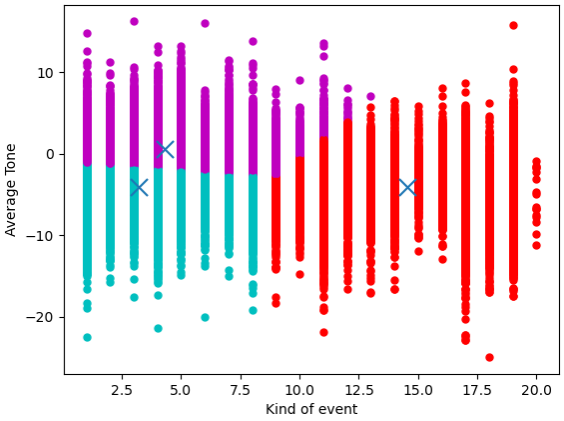
FEBRUARY



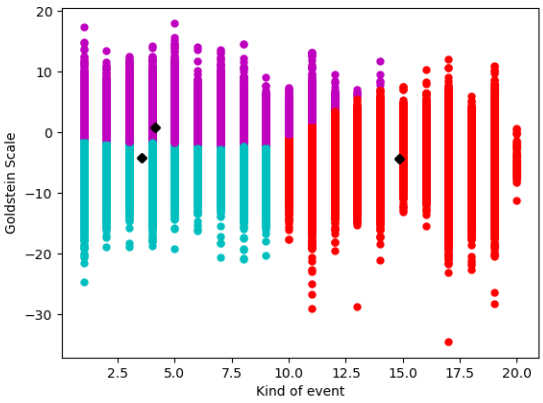
MARCH



APRIL



MAY



JUNE

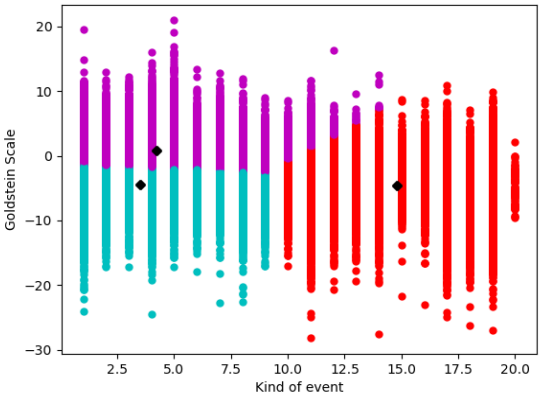


Table 6.1.5. Clustering of all dataset for the measures average tone and kind of event for each of the six months

They have been clustered in 3 groups, considering the optimal number of clusters calculated previously. Despite that the clusters are quite similar in all these months, there are small differences between them. To analyze it, it is necessary to consider the table 6.1.6 which gives us a lot of information.

The first measure is the average value for kind of event. Considering that there are 20 different types of events, the fact that the average value for all months goes from 6.32 to 6.94 which means that there are more events for the first half than the second one. The behavior of the data considering the average tone is similar than the previous measurements; it increases for the months of March and April.

Next, let us analyze the clusters one considering the maximum and minimum values, we can appreciate for each cluster how the numbers increase mainly for the month of April and May, having the best peak for the month of April. In the case of the minimum values, they decrease as the time goes by, having the peaks on May and June.

The following values are the centroids and we can observe how we have the same behavior as the average values. The best values for the centroids are achieved during the months of April and May, although there is not a lot of difference between them. Finally, there is represented the standard deviation for each. Therefore, we get to know how accurate the clusters are.

Another option to cluster the data is in three-dimensional mode, which means that we are able to analyze three inputs at the same time. On the next scheme there is represented this type of clustering in order to find some correlations between type of event Goldstein scale and average tone.

AT – Event Code	K	Nº OF EVENTS	MEAN VALUE		For each cluster							
					MIN VALUES		MAX VALUES		CENTROIDS		STANDARD DEVIATION	
			EC	AT	EC	AT	EC	AT	EC	AT	EC	AT
January	3	4.07M	6.83	-2.32	1	-1.39	12	13.93	4.05	1.29	2.16	2.14
					11	-23	19	9.93	15.04	4.56	3.29	3.44
					1	-20.5	9	-1.94	3.66	-4.45	2.24	2.26
February	3	4.24M	6.65	-2.13	1	-1.33	19	17.02	4.08	1.25	2.17	2.12
					11	-21.05	19	11.15	15	-4.39	3.24	3.55
					1	-25.54	9	-1.99	3.61	-4.37	2.24	2.26
March	3	4.32M	6.37	-2.18	1	-24.24	9	-2.34	3.47	-4.10	2.09	2.08
					11	-22.72	16	11.89	14.96	-3.91	3.20	3.44
					1	-1.34	5	21.43	4.08	0.72	2.15	1.99
April	3	3.60M	6.32	-2.18	1	-1.09	11	13.56	4.29	0.63	2.24	1.91
					11	-21.88	19	15.79	14.53	-4.06	3.43	3.36
					1	-22.5	8	-2.72	3.20	-4.08	2.03	2.09
May	3	3.38M	6.54	-2.22	1	-24.32	9	-2.40	3.54	-4.38	2.28	2.12
					11	-26.47	20	7.14	14.85	-4.25	3.30	3.47
					1	-1.52	12	15.15	4.09	0.72	2.23	1.99
June	3	3.94M	6.94	-2.57	1	-25.19	9	-2.45	3.57	-4.57	2.29	2.16
					11	-22.22	19	9.37	14.81	-4.51	3.23	3.33
					1	-1.63	5	25.37	4.10	0.72	2.25	2.05
ALL DATA SET	3	24M	6.62	-2.27	1	-19.34	9	-2.18	3.57	-4.26	2.24	2.16
					17	-27.34	19	8.95	14.93	-4.26	3.26	3.46
					1	-1.38	14	11.88	4.08	0.94	2.19	2.03

Table 6.1.6. Table with the measures of the clusters from January to June for Goldstein scale and average tone

Another option to cluster the data is in three-dimensional mode, which means that we are able to analyze three inputs at the same time. On the next scheme there is represented this type of clustering in order to find some correlations between type of event, Goldstein scale and average tone.

Let us start with the one cluster that is totally separated from the others. This is composed by the events with the lowest values considering their mentions and the impact on the stability of the country. There is also a correlation with the type of events as this cluster is represented especially by the second half of event codes, from 10 to 20. These include, for instance, events related to disapprove, reject, threaten, protest, fight, assault or violence.

The other two clusters are differentiated by average tone measure. One of them contains the events with positive mentions while the other one represents negative mentions. That means that the types of event from 0 to 10 do not follow any kind of pattern although they seem better considering Goldstein scale. These are related to appeal, cooperate, consult or investigate for example.

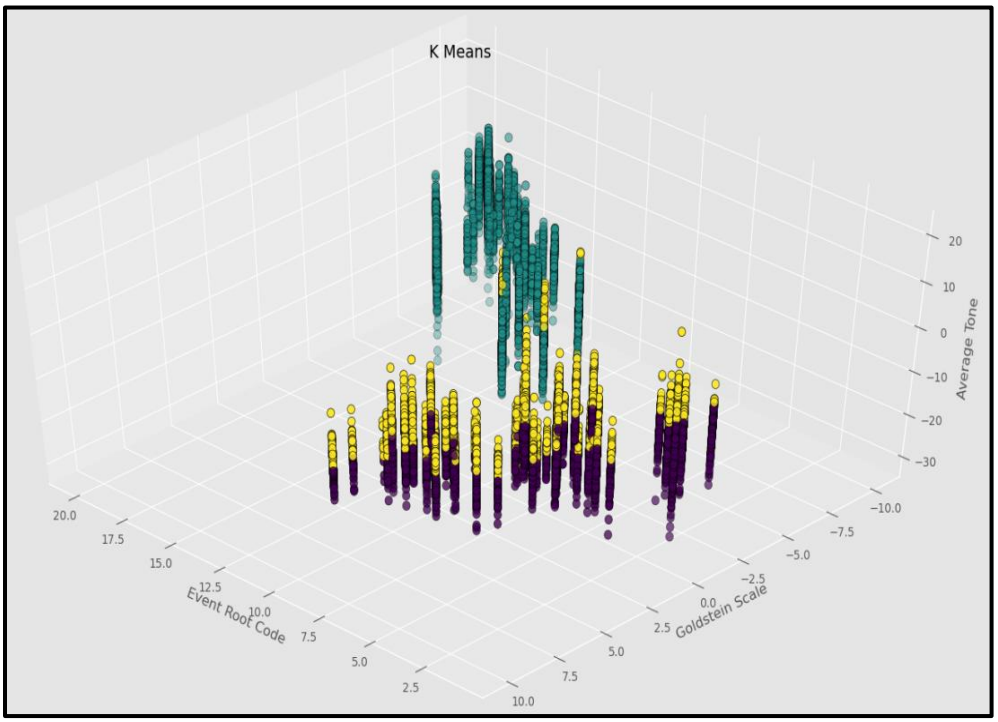


Figure 6.1.5. Three-dimensional clustering

THREE-DIMENSIONAL CLUSTER		Event Code	Golds. Scale	Avg Tone
FOR EACH CLUSTER	MIN VALUES	1	-2	-27.27
		1	-2	0.81
		17	-9	-34.78
	MAX VALUES	8	10	-5.35
		5	3.4	36.36
		17	-5	13.38
	CENTROIDS	2.93	2.11	-4.08
		4.33	3.78	0.78
		14.45	-5.45	-4.13
	STANDARD DEVIATION	1.83	2.12	2.35
		1.96	2.43	2.25
		3.47	2.94	3.50
	MEAN VALUE	6.61	0.66	2.28
	Nº CLUSTERS	3		
	Nº EVENTS	5M		

Table 6.1.7. Table with the measures of the clusters for the measures: kind of event, Goldstein scale, average tone

6.2. Hierarchical clustering

Once the data have been analyzed with the *k-means* algorithm, we are going to repeat the same but using another type of algorithm called agglomerative hierarchical clustering. It is a technique where initially each data point is considered as an individual cluster. Therefore, at each iteration, similar clusters merge with other ones until one cluster or K clusters are formed as it is observed on the figure 6.2.1. This clustering is calculated by taking into account events from January to June and the two measures Goldstein Scale and Average Tone.

From a general point of view, the values for axis ‘x’ cannot be appreciated as there are too many different values and consequently, too many small clusters from the first line.

The orange “cut” associated with the largest gap generates two clusters: the red one and the green one. Therefore, according to this Dendogram the optimal number of clusters should be two.

To sum up, hierarchical clustering is deterministic, which means it is reproducible. However, it is also greedy, which means that it yields local solutions. In other words, this algorithm is appropriate for datasets with a small amount of data so it can be easier to analyze the results.

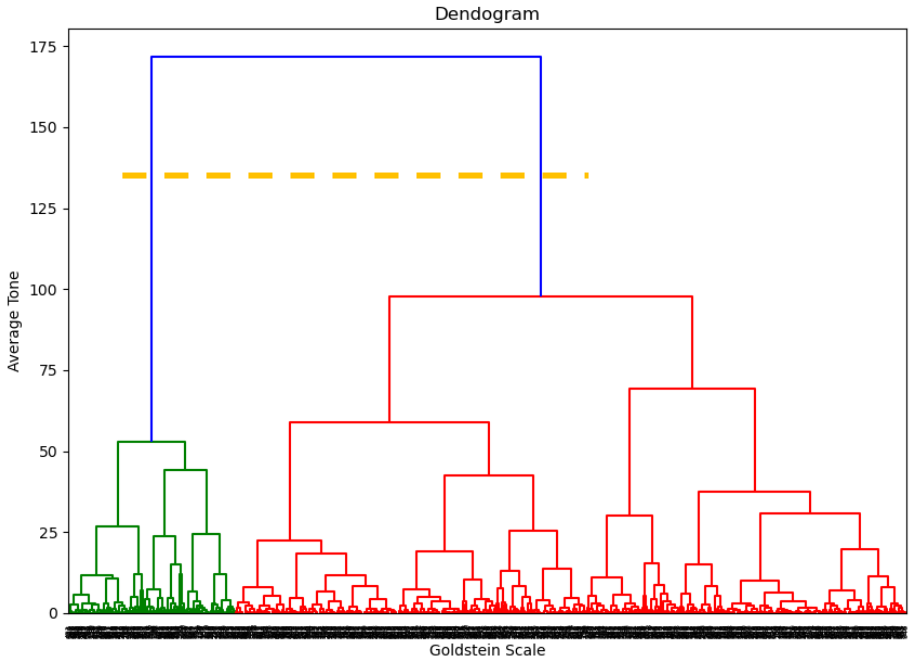


Figure 6.2.1. Dendogram calculated with all the data and the two measures Goldstein Scale and average tone.

7. COUNTRIES ANALYSIS

Once we have a general view of the events, we are going to analyse the countries involved. Subchapter 7.1 includes their behaviour and how they evolve during these six months considering the evolution of the virus. The next subchapter is concentrated on ten selected countries in order to define their evolution, impact or if there is some kind of reciprocity between Actor 1 and Actor 2. The section 7.3 tries to identify some patterns or groups between them. Moreover, the subchapter 7.4 has a similar content as 7.3 but considering all countries instead of just 10. And finally, the section 7.5 analyse some information related to China.

7.1. General view of countries

Let us start with the most common countries for Actor 1 and Actor 2. Remember that the event action is represented as what the first actor did to the second one. The figures 7.1.1 and 7.1.2 represents the evolution of top countries with most events involved for each month from January to June. The first chart represents the countries for Actor 1 and the second one for Actor 2.

On the first one, there are represented five countries. For the first month we have United States, United Kingdom, Iran and China as top countries. According to CNN[8], tensions between the US and Iran hit a boiling point this month, although they have been simmering for decades. Also, the United Kingdom had a lot of importance as it formally left the European Union according to the article 'Brexit Update'[9] written by Gustaf Duhs and Luke Stewart. In this case, we have the same results for Actor 2 which means that these countries were both making and receiving actions.

On February, there is one country that stands out from all the others, except for USA, as we can observe in both figures. This one is China, the origin country

[8] https://edition.cnn.com/interactive/2020/01/world/us-iran-conflict-timeline-trnd/
[9] https://www.stevens-bolton.com/site/insights/articles/brexit-update-august-2020

where the pandemic started. We can suppose that the highest peak is found in this month because it was the time when the propagation of the virus started among other countries. From the other hand, there is another country that appeared which is India. During these months several affairs took place in this country such as the president of United States of America, Donald Trump, visit India by joining the Prime Minister Narendra Modi[10]. Moreover, New Delhi streets turned into battleground, Hindus vs. Muslims, according to The New York Times[11].

For the rest of months, there are not big differences between the three most common countries. However, if we have a look on US it can be appreciated how the events increased during the month of June. It is not a coincidence that is when the country had the most Coronavirus cases in the first semester of the year, according to Worldmeter [12].

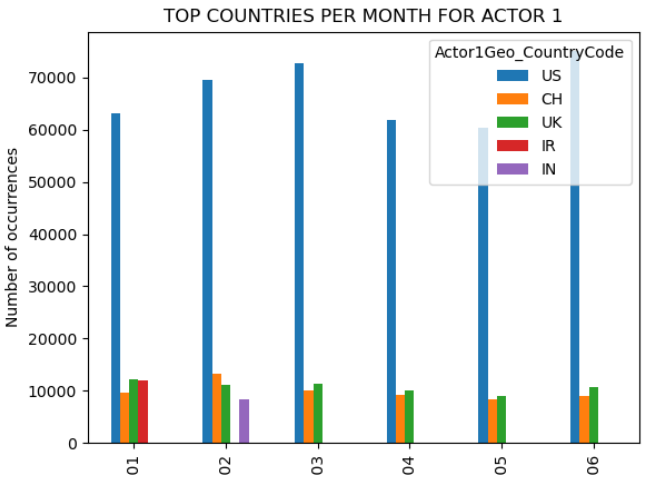


Figure 7.1.2 Top countries with most events per month (Actor 1)

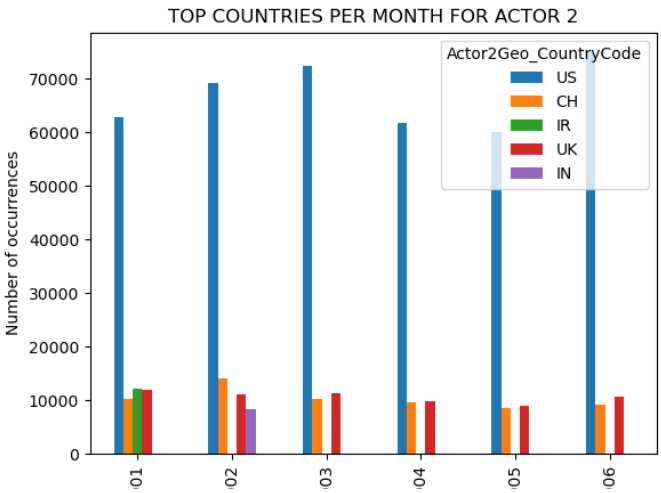


Figure 7.1.2. Top countries with most events per month (Actor 2)

[10] https://www.nytimes.com/2020/02/24/world/asia/trump-india.html
[11] https://www.nytimes.com/2020/02/24/world/asia/trump-india.html

Goldstein Scale

Another way to analyse the data taking into account the countries is Goldstein scale. From -10 to 10, it shows the theoretical potential impact that type of event will have on the stability of a country.

From general perspective, most of the events are located between 0 and 10 on

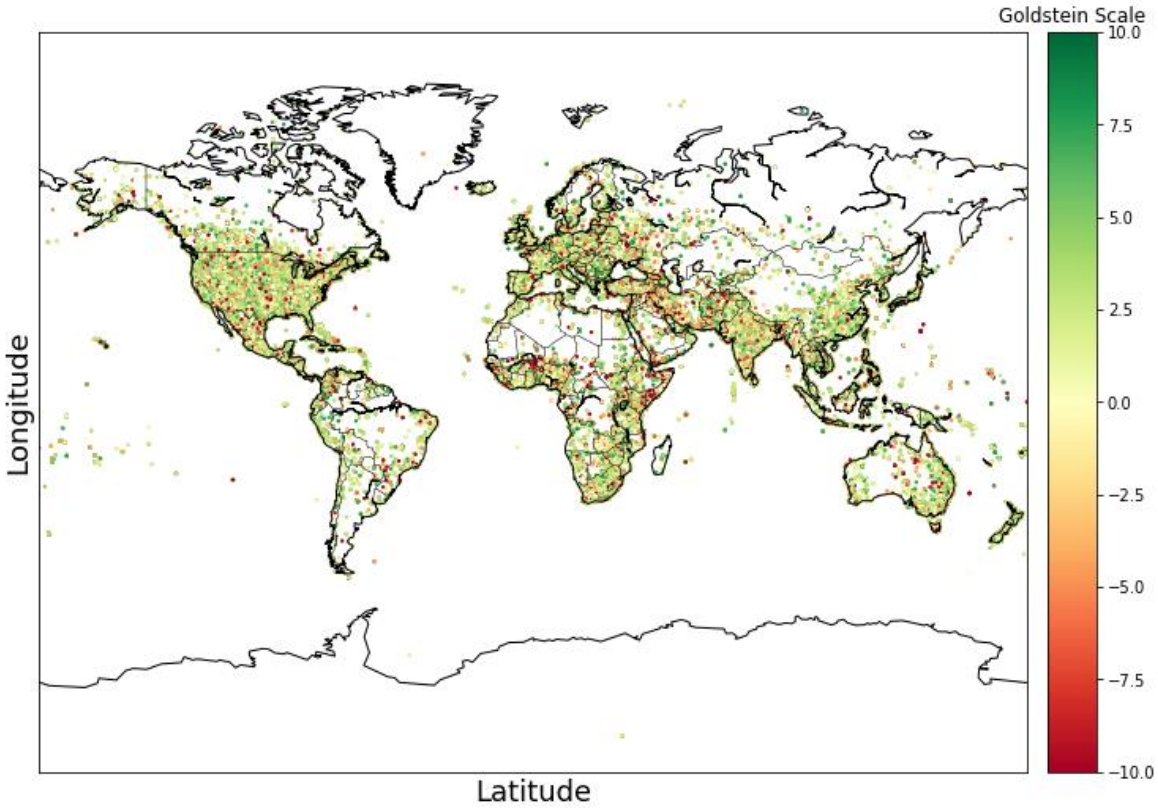


Figure 7.1.3. World map of Goldstein scale

Goldstein scale which means that these ones have a neutral or positive impact. However, there are some events that decreased the stability of their country which are dispersed all over the world although there are some locations that are concentrating an important amount of these negative events. These are located on the continent of Africa near Burkina Faso, Mali and Niger and also between Kenya and Somalia. And on the other side, we can also locate them in the Middle East Asia.

Average Tone

The figure 7.1.4 represents average tone of all documents containing one or more mentions of this event where the score goes typically from -10 (extremely negative) to 10 (extremely positive). In this case, we can consider that more of the half of the events is located between 0 and -10 which means that these are neutral or negative mentions. The locations with more negative mentions are located on the same zones as for the previous analysis with Goldstein scale (some countries of Africa and Middle East Asia). It makes sense if we take into account that these zones are the ones that normally have some type of conflict.

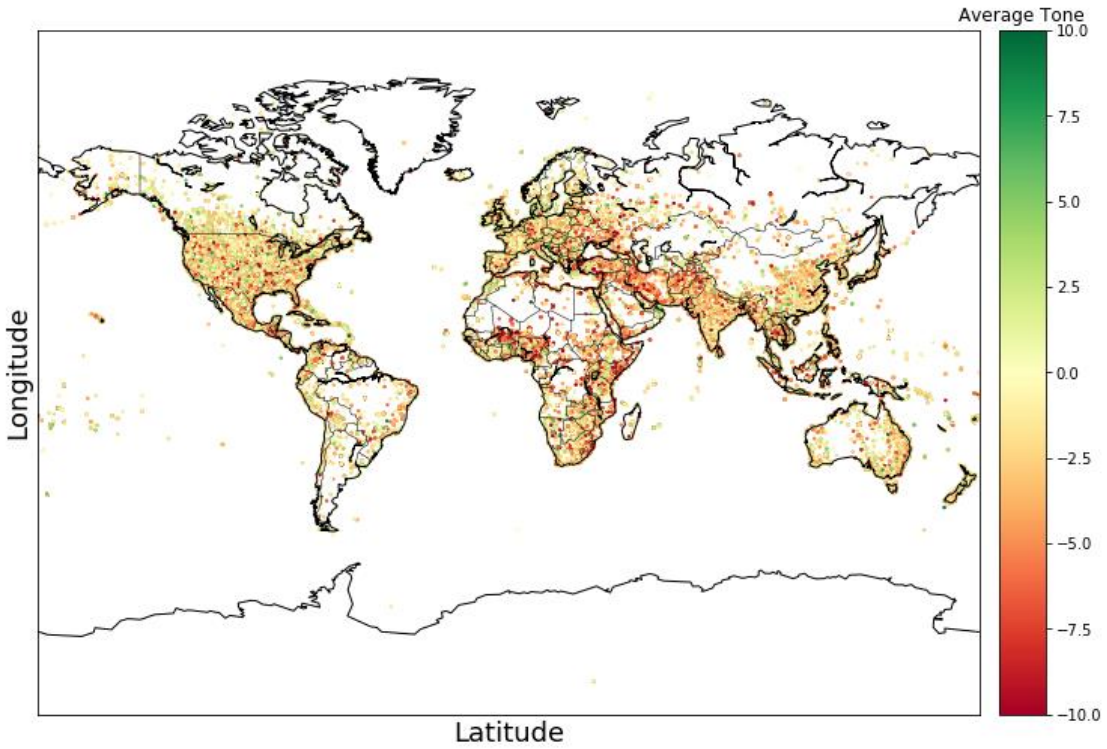


Figure 7.1.4. Average tone represented in a World map.

7.2. Study of ten selected countries

Once all the countries have been analyzed from a general point of view, it would be convenient to select ten countries in order to realize a deeper analysis for them. Their selection has been based on the popularity of the country, the impact of COVID-19 on them and their variety between by comparing the

location. To understand the following graphs is important to consider the proper codes for each country which are showed in brackets: Brazil (BR), China (CN), India (IN), Italy (IT), Poland (PL), Russia (RS), Spain (ES), Turkey (TU), United Kingdom (GB) and United States of America (US).

The main objective of these analysis is getting to know some information such as how stable are the countries; if there are strong relations between these countries considering several measures such as the kind of event; if the relation between the countries changed as the time went by taking into account the spread of the pandemic, etc. Therefore, the first thing that it has to be considered is the evolution of the density of events for each of these countries previously mentioned.

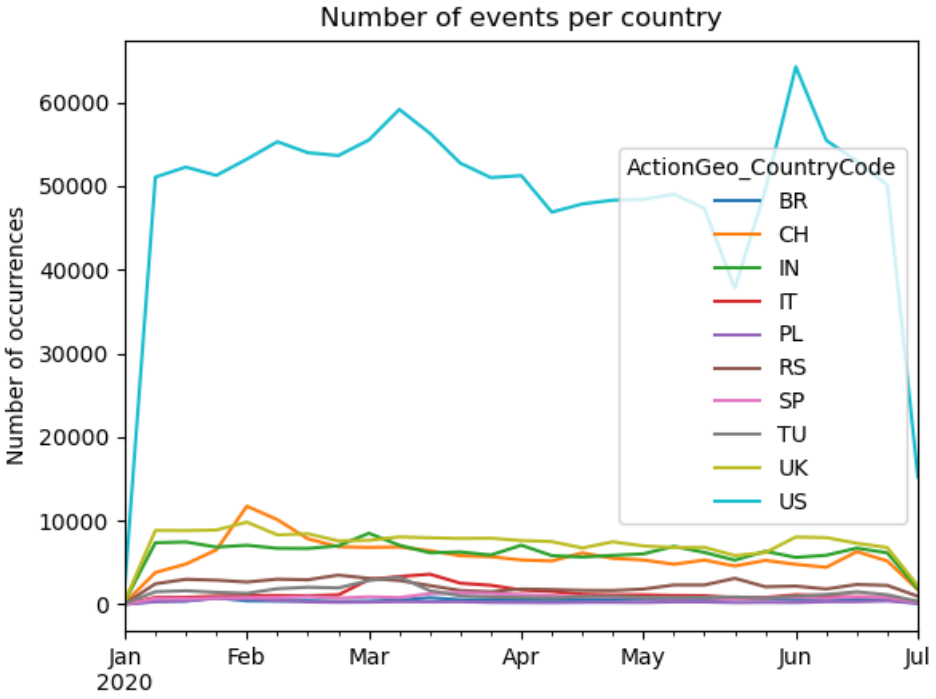


Figure 7.2.1. Number of events for each country

As we can observe at the figure 7.2.1, for the main country which is the United States of America, there is a smooth increase between at the beginning of March. After that, the events decreased considerably until having the highest peak during the month of June.

According to *Worldmeter*[12], a provider of global COVID-19 statistics for many caring people around the world, the statistics are strongly related to the results obtained. This website shows how the cases in this country started appearing at the beginning of March and then, from the month of June there has been a really important increase. Therefore, we can note that the number of general events was affected by the pandemic, which means that it had a lot of influence not only in events related to health but maybe economically and politically as well.

For the rest of the countries, we can appreciate a similar evolution if we have a look on the figure 7.2.2 which is the same figure as the previous one but zoomed for the rest of countries. From January to June, China is the country with the highest peak by far which is located at the beginning of February.

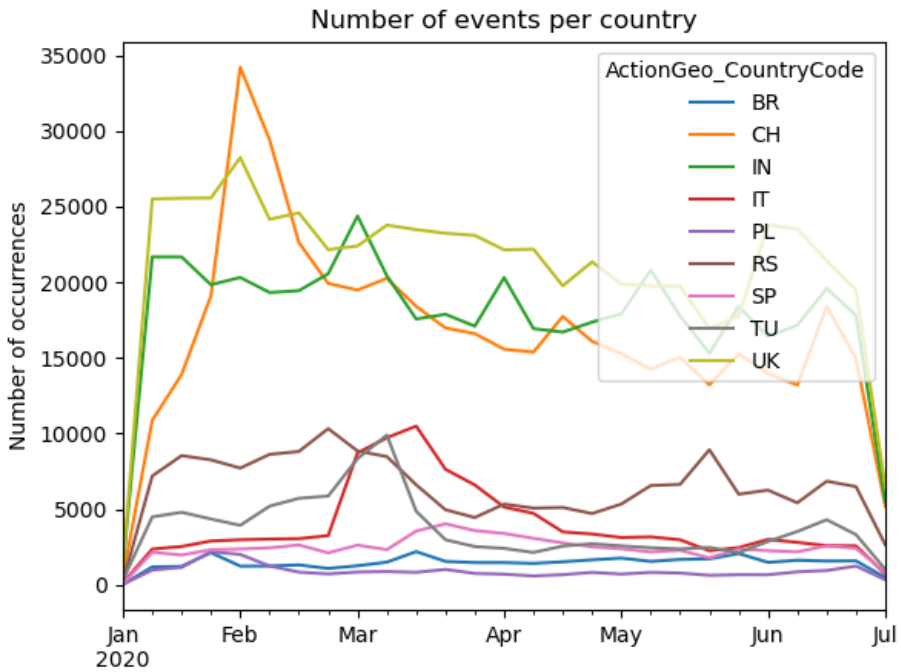


Figure 7.2.2. Zoom of the figure 7.2.1 on the selected countries except US

Almost all the rest of countries share a similar evolution with their highest peak between the second and third month of the year. However, during the month of May, Russia is the only country with a remarkable peak as it is exactly when the cases started increasing a lot all of a sudden, according to Worldmeter.

[12] https://www.worldometers.info/coronavirus/country/us/

Moreover, it is nice to know the average value of Goldstein scale of the events grouped by each of these countries. In the following graph (7.2.3), we can observe this value in an ascending order for each of the country. Regarding the difference between the lowest and highest values, the range is quite small even though it seems the other way.

Turkey and India are the countries with the lowest average although they are not negative but around zero which means that these events had a neutral impact on the stability of their country.

Next, Russia, USA and Brazil have more or less the same value. These countries have one thing in common that may have made decreased the average value on Goldstein scale. As reported by John Haltiwanger[13] on May 26th, the US, Brazil, and Russia had the highest numbers of confirmed coronavirus cases in the world, for that moment.

Then, Poland is a country about which we have a small quantity of information considering the figure 7.2.2. Therefore, according to The New York Times[14], the most important news about Poland that may have caused it have this value are the delay of presidential election due to the virus; the women protest changes to Poland's abortion laws and the locked-down facing closed borders, economic wounds and dire warnings.

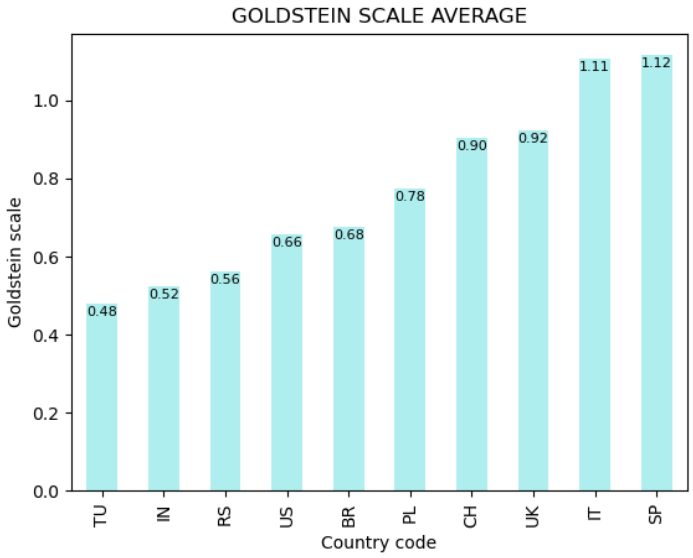


Figure 7.2.3. Average Goldstein scale value for each selected

[13] https://www.businessinsider.com/trump-putin-and-bolsonaro-anti-science-leadership-worst-coronavirus-outbreaks-2020-5?IR=T

[14] https://www.nytimes.com/topic/destination/poland

Then we have China which was the origin of this pandemic as it was mentioned before although they have been able to control the spread within their own country, not like UK, Spain or Italy which have the best values for Goldstein scale although the three of them had the same bad impact considering the virus. However, that may be the main reason why they have very similar values. From a general point of view, it is not what it was expected as they have been the most affected countries from Europe regarding the number of cases or the death rate as well as the economy of the country.

7.2.1. Reciprocity between actor 1 and actor 2

Given a specific country as "*Actor1Geo_CountryCode*", it is good to know the percentage of events and compare it with the percentage for the same country as "*Actor2Geo_CountryCode*" and then, repeat it for the ten countries specified previously. With these results we can appreciate the reciprocity between Actor 1 and Actor 2 or if the countries were the ones who caused the action or, otherwise, the ones who received it.

From a general point of view, there is not much difference between the number of events for Actor 1 and Actor 2. However, we can make a distinction of the countries that acted more as the receivers of the action rather than the cause. These are China, Italy, Poland and Spain.

COUNTRY	% EVENTS FOR ACTOR 1	% EVENTS FOR ACTOR 2
BRAZIL	0.781	0.776
CHINA	9.288	9.675
INDIA	7.019	6.979
ITALY	2.169	2.199
POLAND	0.469	0.474
RUSSIA	3.951	3.924
SPAIN	1.260	1.263
TURKEY	2.356	2.278
UK	10.076	9.986
USA	62.630	62.445

Table 7.2.1. Percentage of events for each country as Actor 1 and Actor 2

7.3. Identification of countries patterns and groups

One of the most important analyses is the clustering considering all countries and also the selected ones. For that, k-means is going to be used as the main clustering algorithm. Therefore, given a vector of two dimensions with these **ten countries**, the measures "GoldsteinScale" and "AvgTone" are going to be used in order to classify and cluster the data depending on the average value of the Goldstein scale and average tone for each country and taking into account the six months.

Moreover, the same cluster is going to be calculated but considering the number of cases per country until June 30th as well as Goldstein scale measure.

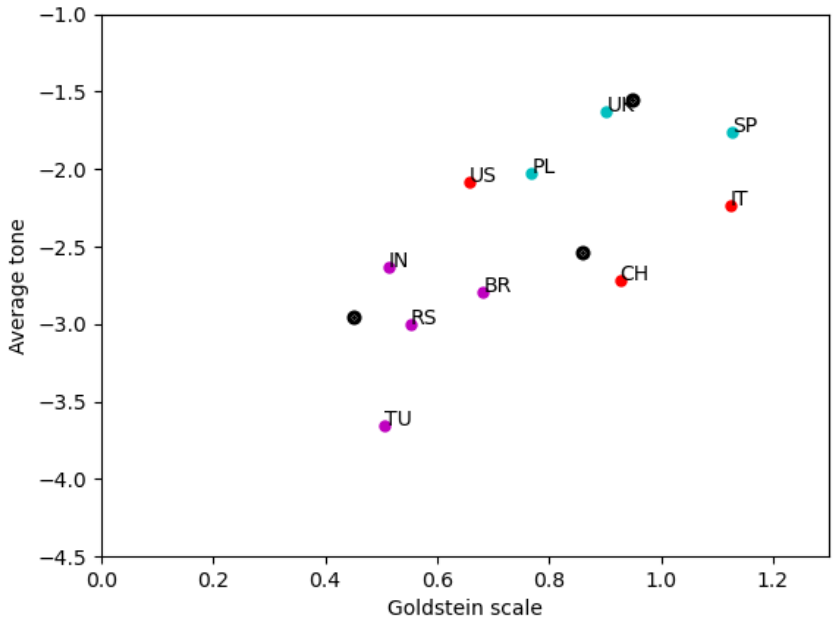


Figure 7.3.4. Countries clustering using the measures of Goldstein scale and average tone

Considering the figure 7.3.4, the countries have been clustered in three different groups. The first one is represented by Brazil, India, Russia and Turkey. The four of them have similar values for average tone although there is some difference between Turkey and the others as its events have a worse impact. Next, we have as a second cluster China, Italy and US.

Finally, the last cluster is defined by Poland, Spain and UK. These are considered as the ones with the best performance for their own country, where United Kingdom has the best value for average tone. That means that it has better mentions although all of countries have a negative average value.

	Golds. Scale	Avg Tone
MIN VALUES	0.51	-3.66
	0.93	-2.71
	0.77	-2.02
MAX VALUES	0.68	-2.80
	1.12	-2.23
	1.13	-1.76
CENTROIDS	0.45	-2.95
	0.86	-2.54
	0.95	-1.55
STANDARD DEVIATION	0.22	0.45
	0.38	0.13
	0.17	0.33
% OF COUNTRIES	40 %	
	30 %	
	30 %	
MEAN VALUES	0.78	-2.45

Table 7.3.1. Values for clustering graph from figure 7.3.4

Next, we can observe the evolution of this clustering month by month considering the two measures called Goldstein scale and average tone.

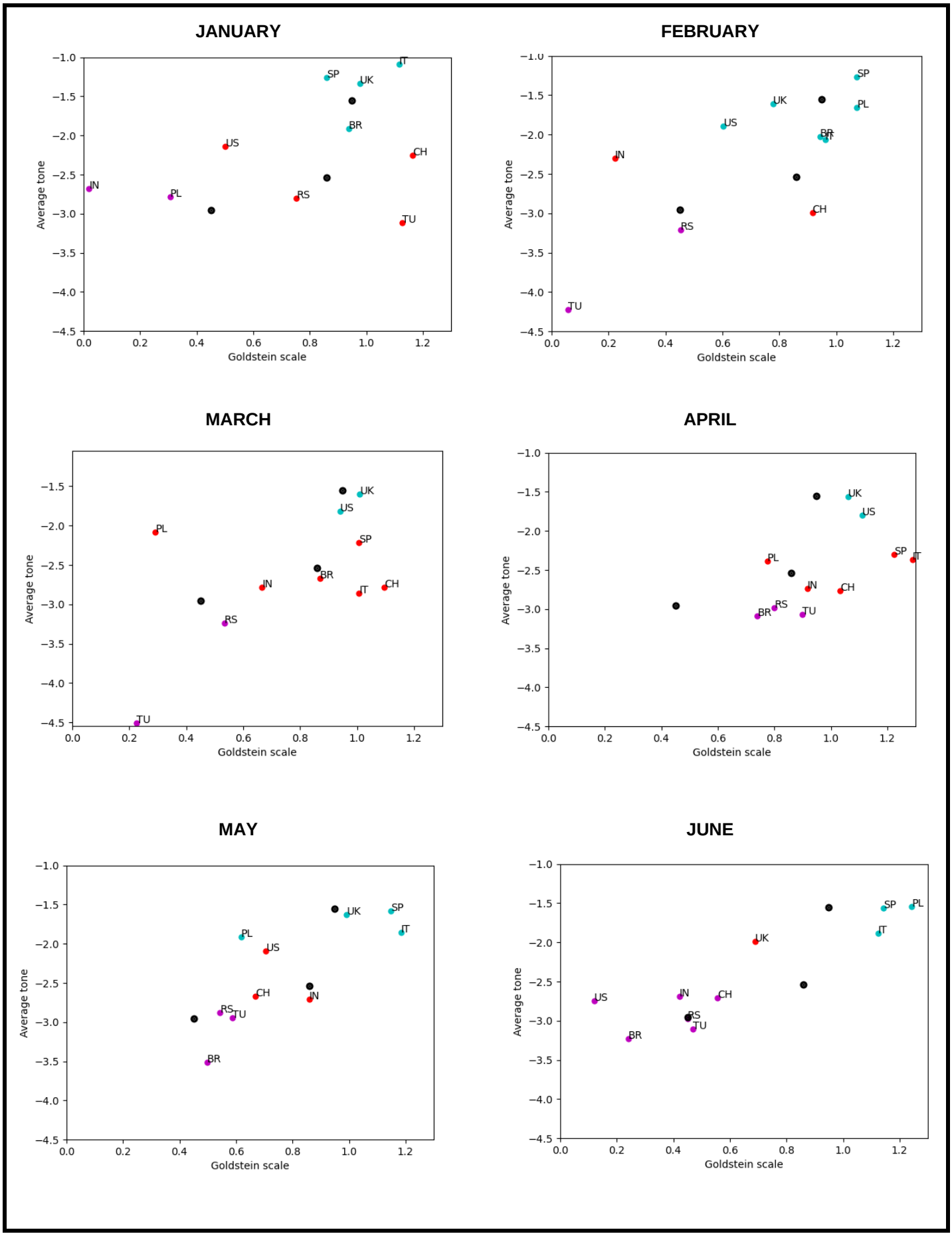


Table 7.3.2. Clustering of ten countries considering Goldstein scale and average tone for every month

MONTH	MEAN VALUE		% EVENTS	MIN VALUES		MAX VALUES		STANDARD DEVIATION	
	GS	AT		GS	AT	GS	AT	GS	AT
January	0.78	-2.14	20	0.02	-2.68	0.31	-2.78	0.20	0.08
			40	0.75	-2.81	1.16	-2.25	0.32	0.46
			40	0.94	-1.91	1.12	-1.09	0.11	0.36
February	0.71	-2.32	20	0.06	-4.22	0.45	-3.21	0.28	0.72
			20	0.22	-2.30	0.92	-2.99	0.49	0.48
			60	0.6	-1.89	1.07	-1.27	0.18	0.30
March	0.76	-2.65	20	0.23	-4.51	0.53	-3.24	0.22	0.90
			60	0.67	-2.78	1.01	-2.21	0.30	0.33
			20	0.94	-1.82	1.01	-1.60	0.05	0,16
April	0.98	-2.51	30	0.74	-3.09	0.90	-3.07	0.08	0.05
			50	0.92	-2.74	1.29	-2.37	0.21	0.22
			20	1.11	-1.80	1.06	-1.56	0.04	0.17
May	0.78	-2.38	30	0.50	-3.52	0.54	-2.88	0.05	0.35
			30	0.67	-2.67	0.71	-2.10	0.10	0.34
			40	0.62	-1.91	1.15	-1.58	0.26	0.16
June	0.76	-2.37	60	0.24	-3.22	0.56	-2.71	0.14	0.24
			10	0.69	-1.98	0.69	-1.98	0.25	0.36
			30	1.12	-1.89	1.24	-1.54	0.12	0.22
Centroids		GS	0.45	0.86	0.95	Nº of clusters		3	
		AT	-2.95	-2.54	-1.55				

Table 7.3.3. Values for the clustering graphs from table 7.3.2 (GS: Goldstein Scale and AT: Average tone)

In every month, the countries are clustered in three groups regarding their similarities. On June, the cluster with worst values for Average Tone is represented by India and Poland. Then, the middle one consists on China, Russia, US and Turkey. And finally, Brazil, Italy, Spain and UK represent the group with the best values. Overall, China and Turkey have the best impact on their country although, on February, their values decreased. According to the reporter Bill Chappell, there were more new infected cases due to Coronavirus

were reported outside China than inside on February 26th, 2020[15]. Moreover, the same day it was informed by the BBC news[16] that the Coronavirus started spreading Europe from Italy.

In addition, Turkey had an important bad impact during the second month of the year. It may be related to Turkey-Syria tensions after several attacks and troops killed, as it is reported by Aljazeera news[17]. For the rest, there are not big differences except for Poland as it had positive mentions on February. However, it had again worse impacts on March following the same pattern as Spain and Italy. As it was previously mentioned, during these month the spread of the virus starting having a big impact especially on European countries.

On April, the range for Goldstein scale is way better than the other months as it can be appreciated on the figure from the table 7.3.2. Therefore, there is a really good increase in almost all countries due to the preventing measures worldwide such as lockdowns, cancellation of big events, etc. However, Brazil had a decrease as there was “A perfect storm” in this country as troubles multiply for the president Bolsonaro who was already struggling to govern effectively when his star minister resigned and accused him of criminal conduct, according to the New York Times[18]. Moreover, the government’s chaotic response to the virus has undercut the country’s ability to cope which made the country have the worst impact on May[19].

During the month of May, the rest of countries remain more or less the same considering average tone values although they had a slightly decrease on Goldstein scale. Nonetheless, US is the country with the worst decrease from April to June, with a similar behavior as Brazil.

During May and June, some important protests took place in the United States of America. As reported by the New York Times, George Floyd, a 46-year-old African-American man, died in Minneapolis on May 25th after being handcuffed

---

[15] https://www.npr.org/sections/goatsandsoda/2020/02/26/809568686/coronavirus-more-new-cases-are-now-reported-outside-china-than-inside?t=1598433558093
[16] https://www.bbc.com/news/world-europe-51638095
[17] https://www.aljazeera.com/news/2020/02/turkey-syria-tensions-escalate-troops-killed-live-updates-200228104334749.html
[18] https://www.nytimes.com/2020/04/25/world/americas/bolsonaro-moro-brazil.html
[19] https://www.nytimes.com/2020/05/16/world/americas/virus-brazil-deaths.html

and pinned to the ground by Derek Chauvin, a white police officer. Consequently, in cities across the United States, tens of thousands of people have swarmed the streets to express their outrage and sorrow descended into nights of unrest, with reports of shootings, looting and vandalism in some cities[20]. Apart from that, new coronavirus cases hit another record in theUS on June, as it is represented on *Worldmeters*[21]. Hence, it is comprehensible the correlation between the negative impact and mentions about the country with those main incidents.

Moreover, India and UK also have been affected in some way during the last month of research as it is reflected in the last graph of the table. Considering once again the information from Worldmeters, the virus started spreading without control in India since June although is not the only fact that may have caused this bad impact on its stability. China and India came to lethal blows due the dispute of some territory which made anger surging in India over deadly border brawl with China as Indians grapple with the deaths of 20 soldiers in chaotic clashes with Chinese troops, as reported by *The New York Times*[22]. On the other hand, UK infected cases decreased from May to June which does not match with the lower values for this country. In spite of this, some other occurrences took place such as a stabbing at UK park declared a 'terrorist incident' as three people died and three were injured as stated in an article from *The New York Times*[23].

To sum up, this year started with a variety of values as each country represented really different behaviors. In the next months, most of them had a more negative impact considering the stability of each country. However, during the month of April and May almost all countries had more positive mentions as well as good balance. And finally, on the last month of the first semester of the year the cluster with the lowest values contains more than the half of the

[20] https://www.nytimes.com/article/george-floyd-protests-timeline.html
[21] https://www.worldometers.info/coronavirus/country/us/
[22] https://www.nytimes.com/2020/06/18/world/asia/india-china-border.html
[23] https://www.nytimes.com/2020/06/21/world/europe/forbury-gardens-stabbing-reading-terrorism.html

countries which means that some bad incidents started occurring related to these regions.

On the figure 7.3.5, there is represented the same countries taking into account Goldstein scale and the number of people infected by the virus until 30th June. Almost all of them are located in the same cluster as the number of cases is similar. However, each one of them had a different impact on their country. Then, Brazil and US represent one cluster each one as they are really different from the others. These have the highest amount of cases although the measures for Goldstein Scale are not as negative as we could suppose.

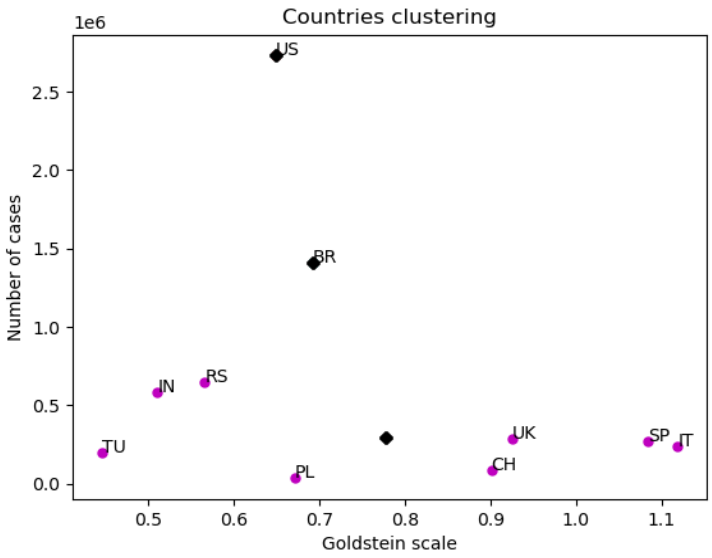


Figure 7.3.5. Countries clustering using the measure Goldstein scale and the number of cases per country until June, 30th

7.3.1. Clustering considering four types of events

Next, we are going to consider the previous ten selected countries in order to analyze them more specifically by contemplating four different scenarios. In each one, there has been selected a type of event which can be interesting and some kind related to the spread of the virus. Therefore, there are represented fours clusters of two dimensions considering Goldstein scale and average tone by calculating their mean value for each country.

Let us start describing which events are representing each one; the graph located on top and left side is defined by the type of events related to providing aid; on its right there are the ones that want to express intent to cooperate; then the graph under these ones at the left side is about requests, proposals, suggestions and appeals and the last one represents the reduce of relations.

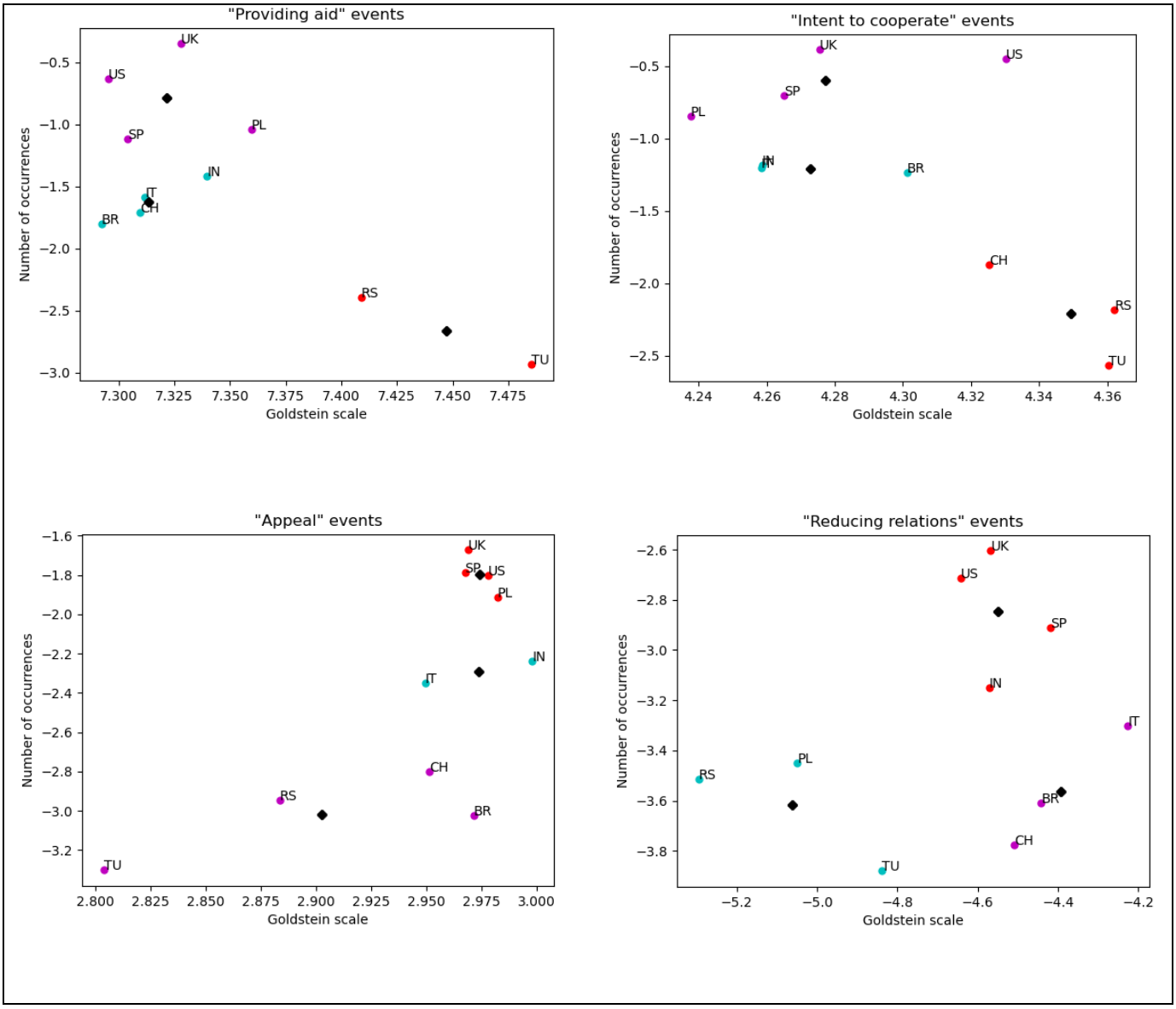


Table 7.3.4 Clustering of the same countries by categorizing the events on four different types.

The first thing to compare is the range for both measures. There is a noticeable difference for Goldstein scale measure between the four graphs. In the first one

all the countries have a really good value as it is about helping by providing any kind of aid. The values are also quite good when it comes to express intent to cooperate which makes sense as any type of union is better than reducing relations, as it is shown in the last graph. That is the reason why it is the only one that has negative values and that is directly connected to a bad impact.

In the middle, there are located neutral events related to appeals where we could say that Turkey is the country with the worst mentions and balance although the difference compared to other countries is really small.

Overall, US, UK, Spain and Poland have similar reactions to the events related to them. Then, Brazil, India, Italy and China are located in same clusters but with more differences between them and finally, Turkey and Russia are always located in the same group. Therefore, there is not any correlation between the countries and their location as they are grouped without regarding from which continent are they from.

7.4. Clustering of all countries

To put these ten countries in context, let us analyze the same vector of two dimensions but with **all countries** where the same measures are going to be used, as it is represented in the figure 7.4.1. Unfortunately, it is very hard to appreciate the graph as the amount of countries is huge. However, on the figure 7.4.2 we can observe the same clustering graph but zooming on the part where the previous ten countries are located.

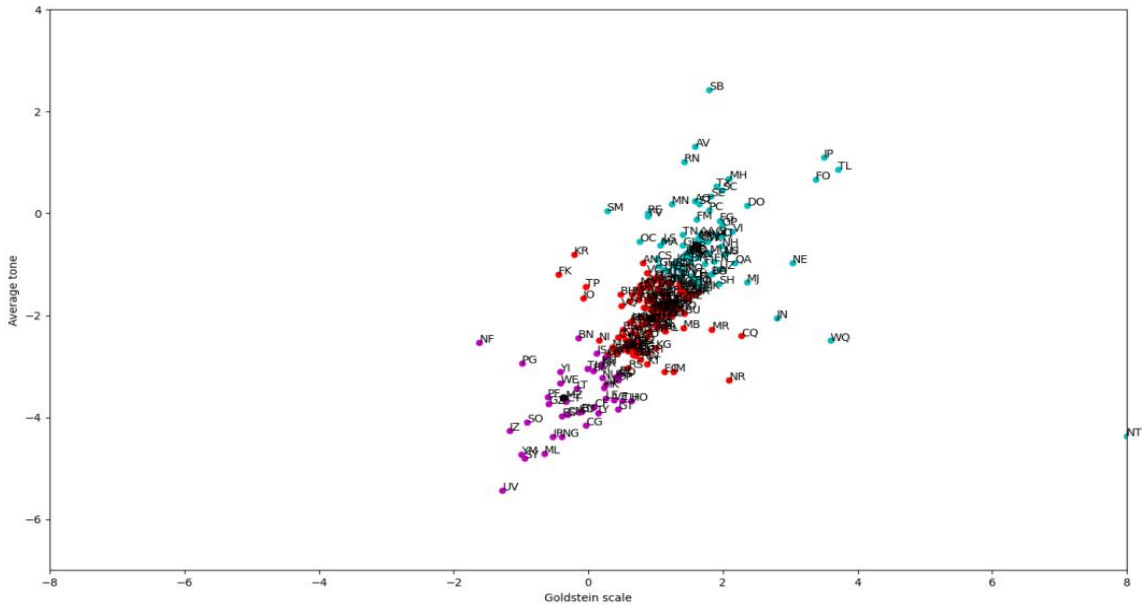


Figure 7.4.1. Countries clustering using the measures of Goldstein scale and average tone

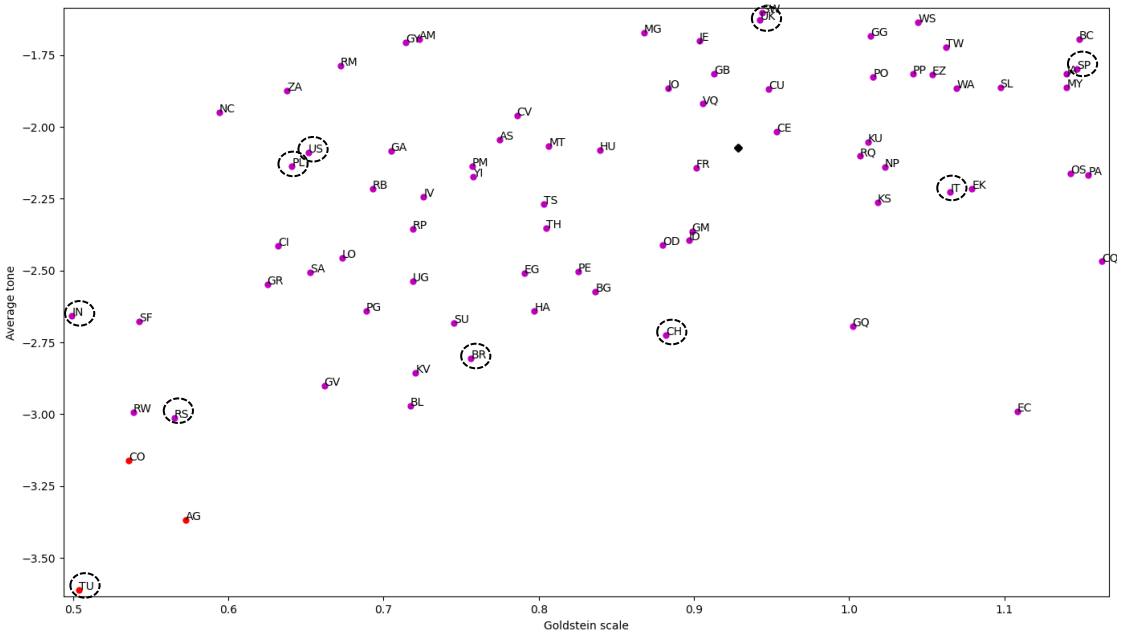


Figure 7.4.2. Countries clustering using the measures of Goldstein scale and average tone (zoomed)

From these ten countries, all of them are located on the same cluster except for Turkey which makes sense as this is the group with the highest number of countries, as we can observe in the table 7.4.8. Therefore, Turkey is placed on the one with the lowest values.

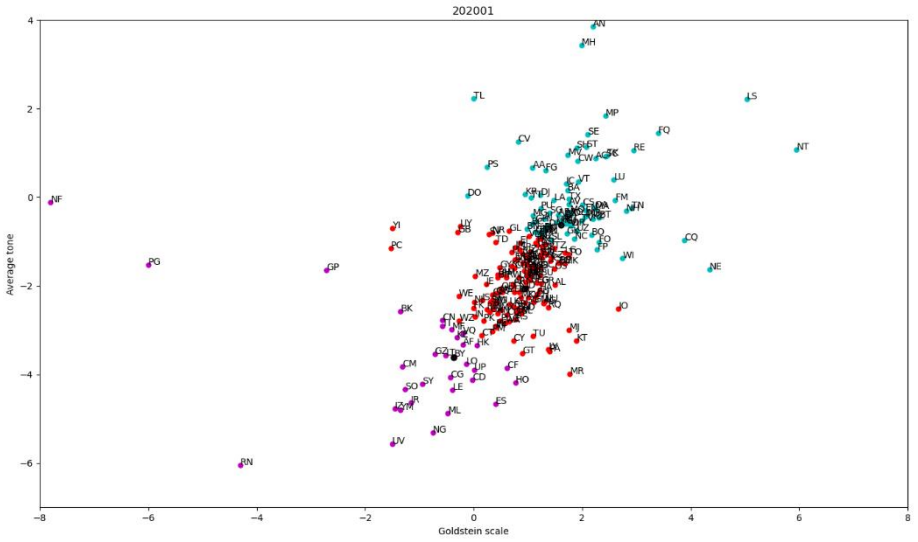
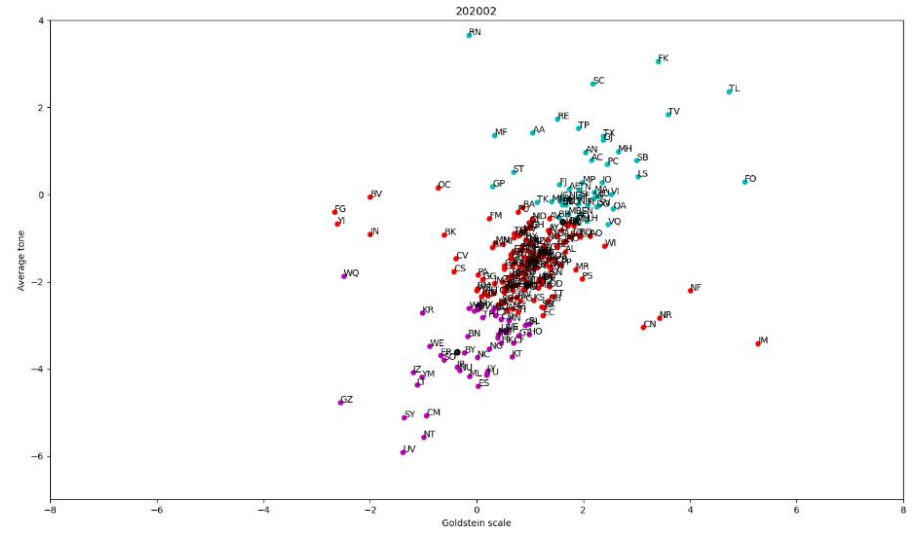
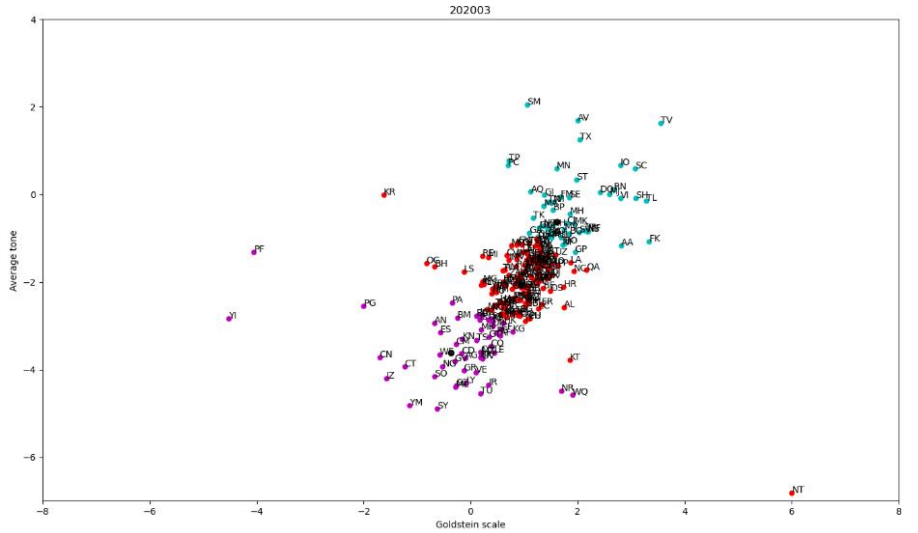
Now, let us have a look on the table 7.4.8, which shows the respective values of the figure 7.4.1. Considering the last row, the mean value for Goldstein scale is positive and that means that the countries rather have neutral-positive impacts on their stability. However, the average tone is negative which represents bad mentions about these countries.

Overall, the middle cluster is the one with the highest percentage of countries as it contains more than the half. Consequently, it is more frequent for a country to have a neutral balance rather than being extremely positive or negative.

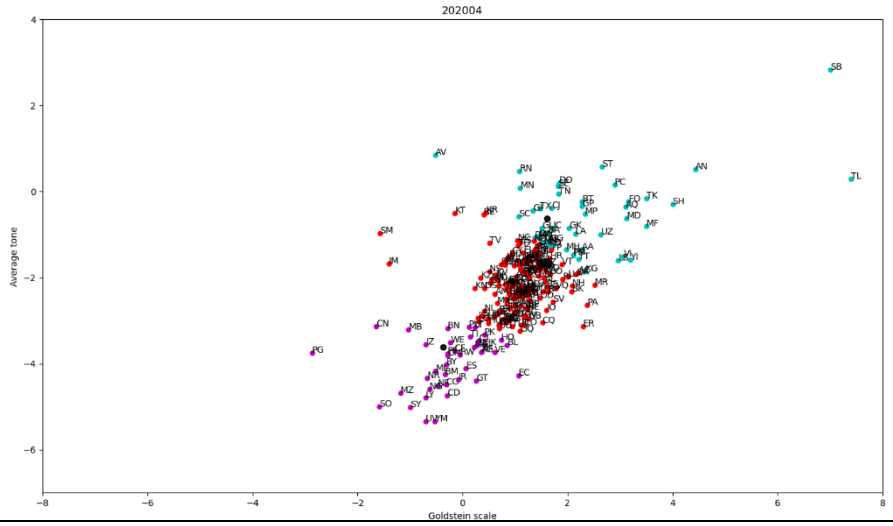
	Golds. Scale	Avg Tone
MIN VALUES	-9.2	0.19
	0.29	-2.77
	1.05	-1.03
MAX VALUES	0.26	-2.81
	1.65	-1.52
	3.50	1.11
CENTROIDS	-0.37	-3.61
	0.93	-2.07
	1.61	-0.62
STANDARD DEVIATION	1.46	0.85
	0.39	0.51
	0.93	0.90
% OF COUNTRIES	17.6	
	51.6	
	30.8	
MEAN VALUES	0.97	-1.84

Table 7.4.1. Values for the clustering of all countries from the figure 7.4.1

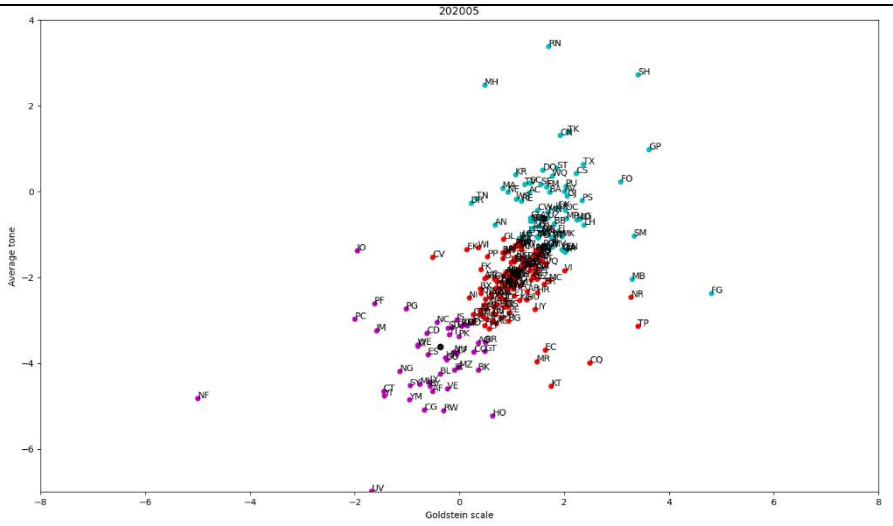
Table 7.4.2. Clustering of all countries considering Goldstein scale and average tone for every month

JANUARY

FEBRUARY

MARCH


APRIL



MAY



JUNE

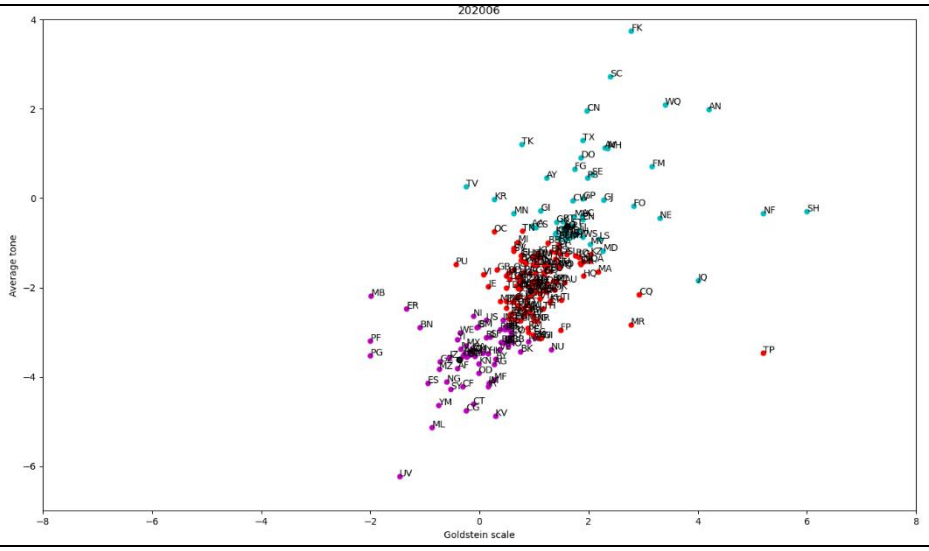


Table 7.4.3. Values for each one of the graph from the table 7.4.2

MONTH	MEAN VALUE		% EVENTS	MIN VALUES		MAX VALUES		STANDARD DEVIATION	
	GS	AT		GS	AT	GS	AT	GS	AT
January	0.89	-1.55	13.46	-5.86	-9.21	0.61	-3.86	1.96	1.52
			53.47	-0.25	-2.78	1.76	-1.30	0.57	0.69
			33.06	-0.12	0.03	5.04	2.21	0.93	1.13
February	0.97	-1.49	18.62	-2.56	-4.77	0.98	-2.97	0.83	0.86
			60.32	-2.61	-0.68	5.28	-3.41	0.90	0.64
			21.05	0.30	0.19	2.80	7.69	0.90	1.40
March	0.93	-1.91	20.73	-4.53	-2.83	0.62	-2.93	1.09	0.70
			58.13	-0.83	-1.58	2.17	-1.72	0.64	0.67
			21.14	1.09	-0.88	2.28	4.06	0.67	0.99
April	1.09	-2.18	17.14	-2.58	-8.73	0.85	-3.57	0.81	0.95
			63.68	-1.40	-1.67	2.52	-2.17	0.53	0.59
			19.18	1.38	-1.08	7.00	2.82	1.35	0.84
May	0.91	-1.93	18.78	-10	-4.14	0.15	-3.12	1.67	0.90
			51.02	1.75	-4.54	3.27	-2.46	0.52	0.64
			30.20	0.67	-0.77	3.40	2.73	0.72	0.98
June	0.96	-1.93	26.83	-1.46	-6.22	1.31	-3.38	0.68	0.67
			51.63	0.55	-2.70	5.20	-3.46	0.60	0.55
			21.54	-0.25	0.26	2.78	3.74	1.08	1.08
Centroids		GS	-0.37	0.93	1.61	Nº of clusters		3	
		AT	-3.61	-2.07	-0.62				

From the table 7.4.2, we can appreciate that, progressively, the countries are less dispersed from January to April. That means that their values were becoming more similar as the time went by. Then, on May and especially on June, these values started dispersing way more than usual.

As it is way hard to make some conclusions from the graphs, let us analyze the values from the table. For each month, there are represented three clusters with

the same centroids for all of them in order to appreciate better the evolution and stability of the countries all over the world.

The cluster represented in purple contains the countries with the lowest values considering Goldstein scale and average tone measures. Taking into account the percentage of the number of countries for every cluster, this is the one with the lowest value. However, there is a small increase from January to March, then after the decrease during April; it increased again especially on June. In other words, during the month of March and April there were less countries that had a negative balance.

The red cluster, with the majority of countries, increased its popularity on February, March and April with a higher percentage of countries. It can mean that countries having worse values became better or, otherwise, the best ones decreased their values. To know that, let us analyze the turquoise cluster. Surprisingly, on January this cluster had the higher amount of countries which means that s third part of them had a really good impact and positive mentions about their country. However, this percentage decreased a lot from February to April, especially the months were the coronavirus had the strongest impact worldwide. Only on May it became as popular as on January, representing 30 % of countries.

Overall, from January to March, more countries started having worse impacts and more negative mentions on average. On April, the countries had really neutral values which are balanced by the best mean values for Goldstein scale and the worst ones for average tone. That means that the countries had a better stability although the mentions were more negative. And finally, on June, the impact of all countries, on average, started behaving the same way as the beginning of the year.

8. DISCUSSION OF RESULTS

After the experimental part, it is nice to make some overall conclusions about the results obtained. GDELT is the database used that contains over twenty-four million events that represent different types of actions, incidents or some phenomenon that happened during the first half of 2020. One of the first aspects we saw is that the number of events increased in February and March and then decreased considerably in April and May. The increment during these first months is related to the spread of the pandemic all over the world by that time. Consequently, there were fewer incidents during the next months due to the prevention measures taken worldwide which made a lot of important events getting canceled.

Next, we can summarize the evolution of the nature of these events. From a general point of view, the year started with many negative incidents such as protests, conflicts between two countries or natural bad phenomenon. At the same time the virus started to appear in the city of Wuhan, China. Therefore, World Health Organization (WHO) announced COVID-19 outbreak as a pandemic on 11th March 2020 because of its highly increase of the spread worldwide. For that reason, the average value of events got better considering their impact on the stability of their country as well as their positive mentions. However, in the last month of the research the performance started getting worse as many countries started coming back to the old pattern.

The last part (*Chapter 7*) is about countries. As it was expected, China has been the leader considering the increment of events related to this country during these months, especially in February. However, other countries showed up in different periods such as Australia, Iran or India, although US and UK are the ones with more presence due to the quantity of its events on this particular database.

Moreover, the evolution of the behavior of these countries has been noticeably affected by the spread of the virus. Some regions such as Spain, Italy and UK

were the ones with more cases in Europe although their performance does not match due to their really good impact on the stability of the country and its positive mentions. However, other countries such as Brazil, India, Russia and US had a worst performance as the number of infected people increased in their country. Nonetheless, those values coincide with other particular incidents related to politics, economy, tensions between countries, protests or natural phenomenon. Overall, February and March had the worst values on average as it was when the virus started spreading the most all over the world and for the same reason, most of the countries had to take preventing measures that made the Earth take a breath.

After all we can say, that because of Coronavirus spread, in some way, the world was more peaceful without negative incidents such as protests, wars or conflicts. However, it had a really negative impact regarding the economy of some countries as well as worldwide health, especially if we consider the death rate.

9. CONCLUSIONS AND FUTURE WORK

Artificial Intelligent is a very important field nowadays as it is able to make our lives easier in different types of environments. In this project, it has been proven that with a large amount of data it is possible to analyze and extract a lot of information by applying methodologies such as data mining, machine learning and complex network analysis.

First of all, I got to realize how important it is to have a theoretical background in order to accomplish a good practical performance. It is essential to understand how the GDELT database is structured to use it in the most optimal way. On the other hand, there were different test scenarios which have been proved to provide useful information for a better understanding which have been the effects of the worldwide pandemic during the first semester of 2020.

About the tools used, there has not been any big problem by using the programs selected. MongoDB [5] as a database is really easy to manipulate by importing, exporting and using the data. Spyder IDE is very practical for data analysis with Python language [4]. However, there has been an important issue related with memory and capacity due to the large amount of data available.

Personally, the development of this project has been really gratifying for several reasons. First of all, I have been able to apply some of the knowledge acquired during my studies such as machine learning and some coding. Also, it is very satisfying to see how all this knowledge related to IT can be applied for the analysis of fields related to politics, economics or health events. Moreover, I had the opportunity to improve myself in AI fields as well as programming skills. And finally, the most grateful fact apart from being able to fulfill the expectations set up at the beginning of the project, is to acquire more information by doing a lot of research about what is happening around the world and especially, how the spread of a virus can change everything.

Future work

In this thesis it has been implemented the basis for the analysis of some data by using different methods to achieve the extraction of useful information. Despite having fulfilled all main expectations for this project, there were planned more analyses which could have given us more interesting conclusions.

One of the possibilities would be evaluating the same practical performance in a more detailed way by considering the evolution of the data day by day, instead of dividing it by months, and considering some other fields which are available in the same database. Moreover, it would be nice to extend the data range by adding these last months of July and August to continue analyzing the evolution of the pandemic.

Another approach would be to compare the statistics from this year with the previous ones, inspect which sectors have been more affected and conclude if it is because of the virus or external causes.

From the technical point of view, it would be nice to upgrade the cluster part by analyzing more algorithms, compare them and conclude which one is more accurate as well as using more dimensions or different type of measures.

The last idea that would have given us a lot of information would be the extension of the analysis related to countries by considering the correlation between their performance on our database and external data such as the GDP per capita, the population density, etc.

To sum up, this project covers the most relevant aspects that could be extracted from the GDELT database where the experiment part proves the implementation of the chapters from the theoretical background. And in addition, there is a brief description of what is yet to come.

BIBLIOGRAPHY

Theoretical part references

[1] "The GDELT Project", [Online] Available: https://www.gdeltproject.org/ (Accessed on 2020.04.06)

[2] "CAMEO Manual 1.1b3", [Online] Available: https://www.gdeltproject.org/data/documentation/CAMEO.Manual.1.1b3.pdf (Accessed on 2020.03.28)

[3] "GDELT Data Format Codebook", [Online] Available: http://data.gdeltproject.org/documentation/GDELT-Data_Format_Codebook.pdf (Accessed on 2020.04.06)

[4] "Python" [Online] Available: https://www.python.org/ (Accessed on 2020.07.27)

[5] "MongoDB: The database for modern applications" [Online] Available: https://www.mongodb.com/ (Accessed on 2020.09.04)

[6] The Economic Times, "Definition of 'Data Mining'", [Online] Available: https://economictimes.indiatimes.com/definition/data-mining (Accessed on 2020.04.06)

[7] Sidath Asiri, "Data Mining in Brief", [Online] Available: https://towardsdatascience.com/data-mining-in-brief-26483437f178 (Accessed on 2020.20.04)

[8] "Machine Learning", [Online] Available: https://cio-wiki.org/wiki/Machine_Learning (Accessed on 2020.07.27)

[9] “What is Machine Learning?”, [Online] Available: https://www.mathworks.com/discovery/machine-learning.html (Accessed on 2020.04.22)

[10] Hunter Heidenreich, “What are types of machine learning?” , [Online] Available: https://towardsdatascience.com/what-are-the-types-of-machine-learning-e2b9e5d1756f

[11] Chaitanya Reddy Patlolla, “Understanding the concept of Hierarchical clustering Technique” [Online] Available: https://towardsdatascience.com/understanding-the-concept-of-hierarchical-clustering-technique-c6e8243758ec (Accessed on 2020.08.03)

[12] Miloš Savić, Mirjana Ivanović, Lakhmi C. Jain, “Fundamentals of Complex Network Analysis”, [Online] Available: https://link.springer.com/chapter/10.1007/978-3-319-91196-0_2 (Accessed on 2020.03.31)

[13] “The R project for Statistical Computing” [Online] Available: https://www.r-project.org/ (Accessed on 2020.07.27)

[14] “Anaconda” [Online] Available: https://www.anaconda.com/ (Accessed on 2020.09.04)

[15] “Pandas” [Online] Available: https://pandas.pydata.org/ (Accessed on 2020.09.04)

[16] “NumPy, the fundamental package for scientific computing with Python” [Online] Available: https://numpy.org/ (Accessed on 2020.09.04)

[17] “Scikit-learn, Machine Learning in Python” [Online] Available: https://scikit-learn.org/ (Accessed on 2020.09.04)

[18] S Joel Franklin, “Elbow method of K-means clustering using Python” [Online] Available: https://medium.com/analytics-vidhya/elbow-method-of-k-means-clustering-algorithm-a0c916adc540 (Accessed on 2020.08.31)

[19] “Working with GDELT in Python: A Quick Tutorial”, [Online] Available: https://blog.gdeltproject.org/working-with-gdelt-in-python-a-quick-tutorial/ (Accessed on 2020.04.06)

[20] Igor Brigadir, “Newsir16-data / Download”, [Online] Available: https://github.com/igorbrigadir/newsir16-data/blob/master/download.py (Accessed on 2020.04.06)

[21] “Python MongoDB Query”, [Online] Available: https://www.w3schools.com/python/python_mongodb_query.asp (Accessed on 2020.04.13)

[22] Colin Bernet, “Fill with Python, read with Pandas”, [Online] Available: https://thedatafrog.com/en/articles/mongodb-python-pandas/ (Accessed on 2020.04.13)

[23] “DataFrame with Pandas”, [Online] Available: https://pandas.pydata.org/pandas-docs/stable/reference/frame.html (Accessed on 2020.04.13)

[24] Felipe, “Pandas Dataframe: Plot Example with Matplotlib and Pyplot”, [Online] Available: http://queirozf.com/entries/pandas-dataframe-plot-examples-with-matplotlib-pyplot (Accessed on 2020.04.20)

[25] COVID-19 coronavirus pandemic [Online] Available: https://www.worldometers.info/coronavirus/ (Accessed on 2020.08.16)

Images references

[1] *MathWorks*, “What is Machine Learning?”, [Online] Available: https://www.mathworks.com/discovery/machine-learning.html

[2] Hunter Heidenreich, “What are types of machine learning?” , [Online] Available: https://towardsdatascience.com/what-are-the-types-of-machine-learning-e2b9e5d1756f

[3] Soler, Carles & Valverde, Anthony & Morales, Castro & Madrigal, Mónica. (2019). Sperm kinematics and morphometric subpopulations analysis with CASA systems: a review. Revista de Biología Tropical. 67. [Online] Available: https://www.researchgate.net/figure/Example-of-agglomerative-and-divisive-hierarchical-clustering-Adapted-from-Everitt-et_fig1_337127088

Appendix A: GitHub repository

The repository GDELT_Analysis is divided into three parts. From one hand, there are all the scripts with the necessary code to reproduce the development part from this project. The first three scripts must be executed to prepare the data. The first one downloads the data from the online available database to a local folder. Then, the second file imports the data from the local folder to the database in MongoDB. And the final script that you must execute before the analysis is the one that cleans the database by removing all the unnecessary files and fields.

Next, the following scripts are useful in order to get all the graphs, plots, charts and tables shown in the development part of this project. Moreover, there is a folder called PLOTS where you can find the results obtained from the previous scripts. These are subdivided into different folders where each one corresponds to one of the scripts.

However, the most important file you must consider the first time you enter to the repository is the file called 'README.md'. It is a guide that explains what it is about and how to use it correctly. For a better understanding, see the Appendix B on the following page.

Appendix B: Guide for the use of the data tools

This guide corresponds to the file “README.md” from the previous repository.

In this repository you will be able to find some scripts about how to use datasets from "The GDELT Project" with Python and MongoDB.

GETTING STARTED

These instructions will allow you to get a copy of the project running on your local machine for development and testing purposes.

Prerequisites

What things do you need to prepare the environment. In the following links there are explained how to install the software and how to use them.

- Python
- MongoDB
- Spyder

Project installation

1. Clone the repository

```
git clone https://github.com/revadhawan/GDELT-Analysis.git
```

2. Install NPM packages

```
npm install
```

Usage

1. Execute the following commands in order to create the database and collection in mongoDB:

```
use GDELT
db.createCollection("Events")
```

2. Download the data from GDELT Project website. For that, execute the following script on Spyder:

- 01_download.py

3. Import the data to the recently database created on mongoDB. For that, execute the following script on Spyder:

- 02_import.py

4. Clean the database to get rid of unnecessary data. For that, execute the following script on Spyder:

- 03_cleanDatabase.py

Once you have the entire environment prepared, you are ready to obtain the graphs shown in this project. For that, you can execute the rest of scripts that are available on the repository.

Developing tools

- Python - Programming language
- MongoDB - The database used
- Spyder - Development environment

Contact

- LinkedIn
- Gmail

Appendix C: Scripts (code)

1. Download

Script '01_download.py' from GDELT-ANALYSIS repository.

```
import os.path
import re
import zipfile
from multiprocessing.pool import ThreadPool
from timeit import default_timer as timer
from urllib.parse import urlparse
from urllib.request import urlretrieve

import pandas as pd

# 01. DEFINE DATA RANGE OF FILES WANTED USING PANDAS
dateRange = pd.date_range(
    start='2020-01-01', end='2020-06-30', freq='1D').tolist()
print("Crawling data from", dateRange[0].strftime(
    '%Y-%m-%d'), "to", dateRange[-1].strftime('%Y-%m-%d'))

# 02. DEFINE GDELT URL AND LOCAL FOLDER URL
gdelturl = "http://data.gdeltproject.org/events/%s.export.CSV.zip"
both_urls = [((gdelturl % ts.strftime('%Y%m%d')),
              "/Users/Reva/Desktop/DATASET02/") for ts in dateRange]

# 03. FUNCTION TO MATCH WEB URL WITH LOCAL URL (LOCAL STORAGE)
def crawl_url(urlFolder):
    url = urlFolder[0]
    folder = urlFolder[1]

    # Use the URL as the name of the file
    filename = folder + os.path.basename(urlparse(url).path)

    if (os.path.isfile(filename)):
        return url, filename, None

    try:
        local_url, http_message = urlretrieve(url, filename)
        return url, local_url, None
    except Exception as e:
        return url, None, e


# 04. FUNCTION TO DOWNLOAD AND EXTRACT FILES
def download_files(both_urls):
    print("Downloading and extracting", len(both_urls), "documents")

    start = timer()
    # Downloading the files (parallel)
    results = ThreadPool(32).imap_unordered(crawl_url, both_urls)

    for url, local_url, error in results:
        if error is None:
            # Extracting the files in a same folder
            z = zipfile.ZipFile(file=local_url, mode='r')
            z.extractall(path='/Users/Reva/Desktop/DATASET02/'+'tmp/')
            print("%r ✔ %.2fs" % (local_url, timer() - start))

```

```
        else:
            print("Error fetching %r: %s" % (url, error))

download_files(both_urls)

print("Finished!")
```

2. Import

Script '02_import.py' from GDELT-ANALYSIS repository.

```
import glob
import pandas as pd
import pymongo
from pymongo import MongoClient

# 01. FIELD NAMES FILE (COLUMN NAMES)
fieldnames = pd.read_excel('CSV.header.fieldids.xlsx',
                           sheet_name='Sheet1', index_col='Column ID')['Field Name']

# 02. LOCATE PATH WHERE FILES HAVE BEEN DOWNLOADED
path = '/Users/Reva/Desktop/DATASET02/tmp'
files = glob.glob(path + "/*.csv")
print(files)

# 03. EXPORT THESE FILES TO MONGO DB
class MongoDB(object):
    def __init__(self, dbName=None, collectionName=None):

        self.dBName = dbName
        self.collectionName = collectionName

        self.client = MongoClient("localhost", 27017)

        self.DB = self.client[self.dBName]
        self.collection = self.DB[self.collectionName]

    def InsertData(self):
        dataframe = []

        for file in files:
            df = pd.read_csv(file, sep='\t', low_memory=False, header=None,
                             dtype=str, names=fieldnames, index_col=['GLOBALEVENTID'])
            dataframe.append(df)
            frame = pd.concat(dataframe, ignore_index=True)
        data = frame.to_dict('records')

        self.collection.insert_many(data)
        print("All the data has been exported to Mongo DB Server")
        print(self.collection.count_documents({}))


# 04. DATABASE AND COLLECTION NAME WHERE IT IS IMPORTED
if __name__ == "__main__":
    mongodb = MongoDB(dbName='GDELT', collectionName='Events')
    mongodb.InsertData()
```

3. Clean database

Script '03_cleanDatabase.py' from GDELT-ANALYSIS repository.

```
1.  import matplotlib.pyplot as plt
2.  import pandas as pd
3.  from matplotlib.pyplot import axis, pie, show
4.  from pymongo import MongoClient
5.
6.  client = MongoClient()
7.  client = MongoClient("mongodb://localhost:27017/")
8.
9.  db = client.GDELT
10. coll = db.Events
11.
12. # 01.CLEANING THE UNWANTED DOCUMENTS
13. coll.delete_many({'Year': {'$nin': ['2020']}})
14.
15. # 02. CLEANING THE UNWANTED FIELDS
16. coll.update_many({}, {'$unset': {'ActionGeo_ADM1Code': 1,
17.                                  'ActionGeo_FeatureID': 1,
18.                                  'ActionGeo_FeautreID': 1,
19.                                  'ActionGeo_FullName': 1,
20.                                  'ActionGeo_Type': 1,
21.                                  'Actor1EthnicCode': 1,
22.                                  'Actor1Geo_ADM1Code': 1,
23.                                  'Actor1Geo_FeatureID': 1,
24.                                  'Actor1Geo_FeautreID': 1,
25.                                  'Actor1Geo_FullName': 1,
26.                                  'Actor1Geo_Type': 1,
27.                                  'Actor1KnownGroupCode': 1,
28.                                  'Actor1Religion1Code': 1,
29.                                  'Actor1Religion2Code': 1,
30.                                  'Actor1Type1Code': 1,
31.                                  'Actor1Type2Code': 1,
32.                                  'Actor1Type3Code': 1,
33.                                  'Actor2EthnicCode': 1,
34.                                  'Actor2Geo_ADM1Code': 1,
35.                                  'Actor2Geo_FeatureID': 1,
36.                                  'Actor2Geo_FeautreID': 1,
37.                                  'Actor2Geo_FullName': 1,
38.                                  'Actor2Geo_Type': 1,
39.                                  'Actor2KnownGroupCode': 1,
40.                                  'Actor2Religion1Code': 1,
41.                                  'Actor2Religion2Code': 1,
42.                                  'Actor2Type1Code': 1,
43.                                  'Actor2Type2Code': 1,
44.                                  'Actor2Type3Code': 1,
45.                                  'DATEADDED': 1,
46.                                  'FractionDate': 1,
47.                                  'IsRootEvent': 1,
48.                                  'NumArticles': 1,
49.                                  'NumMentions': 1,
50.                                  'NumSources': 1,
51.                                  'SOURCEURL': 1,
52.                                  'Year': 1,
53.                                  }})
54.
55. print('Cleaned!')
```

4. General analysis

Script '05_general_analysis.py' from GDELT-ANALYSIS repository.

```
1.  import matplotlib.pyplot as plt
2.  import numpy as np
3.  import pandas as pd
4.  from matplotlib.pyplot import axis, pie, show
5.  from pymongo import MongoClient
6.
7.  client = MongoClient()
8.  client = MongoClient("mongodb://localhost:27017/")
9.
10. db = client.GDELT
11. coll = db.Events
12.
13. events = coll.find(no_cursor_timeout=True)
14. data = list(events)
15. events.close()
16. df = pd.DataFrame(data)
17.
18. # Remove rows with NaN values
19. df = df.replace('null', np.nan, regex=True)
20. df = df[df['Actor1Geo_CountryCode'].notna()]
21. df = df[df['Actor2Geo_CountryCode'].notna()]
22.
23. # 01. NUMBER OF EVENTS PER MONTH
24. df.groupby('MonthYear')['_id'].nunique().plot(kind='bar')
25. plt.title('NUMBER OF EVENTS PER MONTH')
26. plt.xlabel('Month')
27. plt.ylabel('Number of events')
28. show()
29.
30. # 02. NUMBER EVENTS BY THEIR KIND PER MONTH
31. df.groupby(['MonthYear', 'EventRootCode'])[
32.     '_id'].nunique().unstack('MonthYear').plot(kind='bar')
33. plt.title('KIND OF EVENTS PER MONTH')
34. plt.xlabel('Event code')
35. plt.ylabel('Number of occurrences')
36. show()
37.
38. # 03. TOP 5 KIND OF EVENTS WITH MOST FREQUENCY
39. sums = df.groupby('EventRootCode')['_id'].nunique().nlargest(5)
40. pie(sums, labels=sums.index, autopct='%1.1f%%')
41. axis('equal')
42. plt.title('TOP KIND OF EVENTS')
43. plt.xlabel('Event code')
44. plt.ylabel('Number of occurrences')
45. show()
46.
47. # 04. TOP FREQUENT COUNTRIES PER MONTH (ACTOR 1)
48. df.groupby(['MonthYear', 'Actor1Geo_CountryCode'])['_id'].nunique().nlargest(

49.     20).unstack('Actor1Geo_CountryCode').plot.bar()
50. plt.title('TOP COUNTRIES PER MONTH FOR ACTOR 1')
51.
52. # 05. TOP FREQUENT COUNTRIES PER MONTH (ACTOR 2)
53. df.groupby(['MonthYear', 'Actor2Geo_CountryCode'])['_id'].nunique().nlargest(

54.     20).unstack('Actor2Geo_CountryCode').plot.bar()
55. plt.title('TOP COUNTRIES PER MONTH FOR ACTOR 2')
56.
```

```
# Axis names
plt.ylabel('Number of occurrences')
plt.xlabel('Month')
show()
```

5. Basemap

Script '06_basemap.py' from GDELT-ANALYSIS repository.

```
import os

import conda
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
from mpl_toolkits.basemap import Basemap
from pymongo import MongoClient

conda_file_dir = conda.__file__
conda_dir = conda_file_dir.split('lib')[0]
proj_lib = os.path.join(os.path.join(conda_dir, 'share'), 'proj')
os.environ["PROJ_LIB"] = proj_lib

os.environ["PROJ_LIB"] = "D:\\Anaconda\Library\share"

fig = plt.figure(figsize=(12, 9))

client = MongoClient()
client = MongoClient("mongodb://localhost:27017/")
db = client.GDELT
coll = db.Events

# FIND ALL EVENTS
events = coll.find(no_cursor_timeout=True)
data = list(events)
events.close()
df = pd.DataFrame(data)

# REMOVE ROWS WITH NAN VALUES
df = df.replace('null', np.nan, regex=True)
df = df[df['ActionGeo_CountryCode'].notna()]
df = df[df['ActionGeo_Long'].notna()]
df = df[df['ActionGeo_Lat'].notna()]

m = Basemap(projection='mill',
            llcrnrlat=-90,
            urcrnrlat=90,
            llcrnrlon=-180,
            urcrnrlon=180,
            resolution='c')

m.drawcoastlines()
m.drawcountries(color='black')

# DEFINE LATITUD AND LONGITUDE VALUES
lat_x = pd.to_numeric(df['ActionGeo_Long'], errors='coerce').tolist()
long_y = pd.to_numeric(df['ActionGeo_Lat'], errors='coerce').tolist()

# 01.GOLDSTEIN SCALE
goldstein = pd.to_numeric(df['GoldsteinScale']).tolist()
m.scatter(lat_x, long_y, latlon=True, c=goldstein, s=2, cmap='RdYlGn')
bar = m.colorbar()
```

```
bar.ax.set_title('Goldstein Scale')

# 02. GOLDSTEIN SCALE AVERAGE
df['GoldsteinScale'] = df['GoldsteinScale'].astype(float)
y = list(df.groupby('ActionGeo_CountryCode')['GoldsteinScale'].mean())
m.scatter(lat_x, long_y, latlon=True, c=y, s=2, cmap='RdYlGn')
bar = m.colorbar()
bar.ax.set_title('Goldstein Scale')

# 02. AVERAGE TONE
avgtone = pd.to_numeric(df['AvgTone']).tolist()
m.scatter(lat_x, long_y, latlon=True, c=avgtone,
          s=2, cmap='RdYlGn', vmin=-10, vmax=10)
bar = m.colorbar()
bar.ax.set_title('Average Tone')

# 03. EVENT ROOT CODE
event = pd.to_numeric(df['EventRootCode']).tolist()
m.scatter(lat_x, long_y, latlon=True, c=event, s=2, cmap='RdYlGn')
bar = m.colorbar()
bar.ax.set_title('Event Root Code')

plt.xlabel('Latitude', fontsize=18)
plt.ylabel('Longitude', fontsize=18)
plt.show()
```

6. General clustering

Script '07_general_clustering.py' from GDELT-ANALYSIS repository.

```
from fractions import Fraction as fr
from statistics import stdev

import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import scipy.cluster.hierarchy as sch
import seaborn as sb
from matplotlib.pyplot import axis, pie, show
from mpl_toolkits.mplot3d import Axes3D
from pymongo import MongoClient
from scipy import stats
from scipy.cluster.hierarchy import dendrogram, linkage
from sklearn.cluster import AgglomerativeClustering, KMeans

client = MongoClient()
client = MongoClient("mongodb://localhost:27017/")

db = client.GDELT
coll = db.Events

# FIND EVENTS
events = coll.find(no_cursor_timeout=True).limit(20)
data = list(events)
events.close()
df = pd.DataFrame(data)

# Remove rows with NaN values
```

```
df = df.replace('null', np.nan, regex=True)
df = df[df['AvgTone'].notna()]
df = df[df['GoldsteinScale'].notna()]

# 01. INPUTS
v1 = pd.to_numeric(df['EventRootCode'], errors='coerce').values
v2 = pd.to_numeric(df['GoldsteinScale'], errors='coerce').values
v3 = pd.to_numeric(df['AvgTone'], errors='coerce').values

x1 = np.array(v1)
x2 = np.array(v2)
x3 = np.array(v3)

# 02. CREATE COORDINATES WITH THE INPUTS (x,y)
# X = np.array(list(zip(x1, x2)))
X = np.array(list(zip(x1, x2, x3)))
X.shape

# 03. CALCULATE OPTIMAL NUMBER OF K (CLUSTERS)
Nc = range(1, 10)
kmeans = [KMeans(n_clusters=i) for i in Nc]
kmeans
score = [kmeans[i].fit(X).score(X) for i in range(len(kmeans))]
score
plt.plot(Nc, score)
plt.grid()
plt.xlabel('Number of Clusters')
plt.ylabel('Score')
plt.title('Elbow Curve')
plt.show()

# 04. ASSIGNING NUMBER OF CLUSTERS FOR KMEANS
kmeans = KMeans(n_clusters=3)
kmeans = kmeans.fit(X)
labels = kmeans.predict(X)
centroids = kmeans.cluster_centers_

colors = ["m.", "r.", "c.", "y.", "b."]

# HIEARCHICAL CLUSTERING
# dend = sch.dendrogram(sch.linkage(X, method='ward'))
# plt.title('Dendogram')
# plt.show()

# K-MEANS CLUSTERING
# 05. SEPARATING VALUES FOR DIFFERENT CLUSTERS


def groupby(X, labels):
    sidx = labels.argsort(kind='mergesort')
    X_sorted = X[sidx]
    labels_sorted = labels[sidx]

    cut_idx = np.flatnonzero(
        np.r_[True, labels_sorted[1:] != labels_sorted[:-1], True])

    out = [X_sorted[i:j] for i, j in zip(cut_idx[:-1], cut_idx[1:])]
    return out


result = groupby(X, labels)

# 06. CALCULATIONS FOR EACH CLUSTER
for cluster in result:
    for i in range(len(cluster)):
```

```
        sums = cluster[i][0] + cluster[i][1]

    # 2 dimensions
    # print('Min values:', min(cluster, key=lambda x: x[0] + x[1]))
    # print('Max values:', max(cluster, key=lambda x: x[0] + x[1]))
    # print('Standard Deviation for x:', (stdev(cluster[:, 0])))
        # print('Standard Deviation for y:', (stdev(cluster[:, 1])))

        # 3 dimensions
        print('Max values:', max(cluster, key=lambda x: x[0] + x[1] + x[2])
    )
        print('Min values:', min(cluster, key=lambda x: x[0] + x[1] + x[2])
    )
        print('Standard Deviation for x:', (stdev(cluster[:, 0])))
        print('Standard Deviation for y:', (stdev(cluster[:, 1])))
        print('Standard Deviation for z:', (stdev(cluster[:, 2])))

    # AVERAGE VALUE
    print('Mean x:', np.mean(x1))
    print('Mean y:', np.mean(x2))
    print('Mean y:', np.mean(x3))

    # CENTROIDS VALUES
    print("Centroids:", centroids)

    # 07. PLOTTING THE CLUSTERS FOR 2 DIMENSIONS
    for i in range(len(X)):
        plt.plot(X[i][0], X[i][1], colors[labels[i]], markersize=10)
    plt.scatter(centroids[:, 0], centroids[:, 1], marker='x',
                s=20, linewidths=5, zorder=10, color='k')
    plt.xlabel('Kind of event')
    plt.ylabel('Average tone')
    plt.show()

    # 07. PLOTTING THE CLUSTERS FOR 3 DIMENSIONS
    fig = plt.figure(1, figsize=(7, 7))
    ax = Axes3D(fig, rect=[0, 0, 0.95, 1], elev=48, azim=134)
    ax.scatter(X[:, 0], X[:, 1], X[:, 2],
               c=labels.astype(np.float), edgecolor="k", s=50)
    ax.set_xlabel("Event Root Code")
    ax.set_ylabel("Goldstein Scale")
    ax.set_zlabel("Average Tone")
    plt.title("K Means", fontsize=14)
    show()
```

7. Countries analysis

Script '08_countries_analysis.py' from GDELT-ANALYSIS repository.

```
import os
from fractions import Fraction as fr
from statistics import stdev

import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import scipy.cluster.hierarchy as sch
import seaborn as sb
from matplotlib.pyplot import axis, pie, show
from pymongo import MongoClient
```

```
from scipy import stats
from scipy.cluster.hierarchy import dendrogram, linkage
from sklearn.cluster import AgglomerativeClustering, KMeans

# CONNECTION WITH MONGODB
client = MongoClient()
client = MongoClient("mongodb://localhost:27017/")

db = client.GDELT
coll = db.Events

# FIND EVENTS
events = coll.aggregate([{'$match': {'ActionGeo_CountryCode': {'$in': [
                          'BR', 'CH', 'IN', 'IT', 'PL', 'RS', 'SP', 'TU', 'UK',
'US']}}}], allowDiskUse=True)
data = list(events)
events.close()
df = pd.DataFrame(data)

# Remove rows with NaN values
df = df.replace('null', np.nan, regex=True)
df = df[df['AvgTone'].notna()]
df = df[df['GoldsteinScale'].notna()]
df = df[df['ActionGeo_CountryCode'].notna()]

# 01. EVOLUTION OF EVENTS DENSITY
df['SQLDATE'] = pd.to_datetime(df['SQLDATE'])
df.groupby([pd.Grouper(key='SQLDATE', freq='W'), 'ActionGeo_CountryCode'])[
    '_id'].nunique().unstack('ActionGeo_CountryCode').plot()
plt.title('Number of events per country')
plt.ylabel('Number of occurrences')
plt.xlabel('Time')
show()

# 01. GS AVERAGE
v1 = pd.Categorical(df['ActionGeo_CountryCode']).codes
x = pd.to_numeric(df['EventRootCode']).values
df['gs'] = pd.to_numeric(df['GoldsteinScale']).values

df['GoldsteinScale'] = df['GoldsteinScale'].astype(float)
y = df.groupby('ActionGeo_CountryCode')['GoldsteinScale'].mean()
x = list(df.groupby('ActionGeo_CountryCode').groups.keys())
print(x, y)
plot = df.groupby('ActionGeo_CountryCode')['GoldsteinScale'].mean(
).sort_values().plot(kind='bar', color='paleturquoise')

for p in plot.patches:
    plot.annotate(format(p.get_height(), '.2f'),
                  (p.get_x() + p.get_width() / 2., p.get_height()),
                  ha='center', va='center',
                  size=8,
                  xytext=(0, -5),
                  textcoords='offset points')

plt.title('GOLDSTEIN SCALE AVERAGE')
plt.ylabel('Goldstein scale')
plt.xlabel('Country code')
plt.show()

# ACTOR 1
events = coll.aggregate([{'$match': {'Actor2Geo_CountryCode': {'$in': [
                          'BR', 'CH', 'IN', 'IT', 'PL', 'RS', 'SP', 'TU', 'UK',
'US']}}}], allowDiskUse=True)
data = list(events)
events.close()
```

```
df = pd.DataFrame(data)

# Remove rows with NaN values
df = df.replace('null', np.nan, regex=True)
df = df[df['Actor1Geo_CountryCode'].notna()]

# Actor 1
count = df.groupby('Actor1Geo_CountryCode')['_id'].count()
total = sum(count)
x = list(df.groupby('Actor1Geo_CountryCode').groups.keys())
y = count * 100 / total
print(y)

# ACTOR 2
events = coll.aggregate([{'$match': {'Actor2Geo_CountryCode': {'$in': [
                        'BR', 'CH', 'IN', 'IT', 'PL', 'RS', 'SP', 'TU', 'UK',
'US']}}}], allowDiskUse=True)
data = list(events)
events.close()
df = pd.DataFrame(data)

# Remove rows with NaN values
df = df.replace('null', np.nan, regex=True)
df = df[df['Actor2Geo_CountryCode'].notna()]

count = df.groupby('Actor2Geo_CountryCode')['_id'].count()
      x1 = list(df.groupby('Actor2Geo_CountryCode').groups.keys())
      total = sum(count)
      y1 = count * 100 / total
      print(y1)
```

8. Countries clustered

Script '09_countries_clustering_all.py' from GDELT-ANALYSIS repository.

```
from statistics import stdev

import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
from pymongo import MongoClient
from sklearn.cluster import KMeans

# CONNECTION WITH MONGODB
client = MongoClient()
client = MongoClient("mongodb://localhost:27017/")

db = client.GDELT
coll = db.Events

# FIND EVENTS
events = coll.find(no_cursor_timeout=True)
data = list(events)
events.close()
df = pd.DataFrame(data)
```

```
21.
22. # Remove rows with NaN values
23. df = df.replace('null', np.nan, regex=True)
24. df = df[df['ActionGeo_CountryCode'].notna()]
25.
26. # 01. Goldstein scale and average tone
27. df['x'] = pd.to_numeric(df['GoldsteinScale']).values
28. df['y'] = pd.to_numeric(df['AvgTone']).values
29. df = df.filter(['ActionGeo_CountryCode','x','y'], axis=1)
30. df = df.groupby('ActionGeo_CountryCode').mean()
31.
32. # 02. Goldstein scale and number of cases
33. df['x'] = pd.to_numeric(df['GoldsteinScale']).values
34. df = df.filter(['ActionGeo_CountryCode','x'], axis=1)
35. df = df.groupby('ActionGeo_CountryCode').mean()
36. df['y'] = [1408485, 83531, 585792, 240599, 34393, 647849, 272829, 199906, 2832
    53, 2729470]
37. print(df)
38.
39. # MEASURES (INPUTS)
40. x1 = np.array(df['x'])
41. x2 = np.array(df['y'])
42. n = list(df.groupby('ActionGeo_CountryCode').groups.keys())
43.
44. # CREATE COORDINATES WITH THE INPUTS (x,y)
45. X = np.array(list(zip(x1, x2)))
46. X.shape
47.
48. # Centroids for ten selected countries
49. centroids = np.array([[0.45,-2.95], [0.86,-2.54], [0.95,-1.55]], np.float64)
50. # Centroids for all countries (commment if using ten countries)
51. centroids = np.array([[-0.37,-3.61], [0.93, -2.07], [1.61,-
    0.62]], np.float64)
52.
53. # ASSIGNING NUMBER OF CLUSTERS FOR KMEANS
54. kmeans = KMeans(n_clusters=3, init=centroids, n_init=1)
55. kmeans = kmeans.fit(X)
56. labels = kmeans.predict(X)
57. colors=["m.", "r.", "c.", "y.", "b."]
58.
59. # 01. SEPARATING VALUES FOR DIFFERENT CLUSTERS
60. def groupby(X, labels):
61.     sidx = labels.argsort(kind='mergesort')
62.     X_sorted = X[sidx]
63.     labels_sorted = labels[sidx]
64.
65.     cut_idx = np.flatnonzero(np.r_[True, labels_sorted[1:] != labels_sorted[:-
    1], True])
66.
67.     out = [X_sorted[i:j] for i,j in zip(cut_idx[:-1],cut_idx[1:])]
68.     return out
69. result = groupby(X, labels)
70.
71. # 02. CALCULATIONS FOR EACH CLUSTER
72. for cluster in result:
73.     for i in range(len(cluster)):
74.         sums = cluster[i][0] + cluster[i][1]
75.     print('Percentage of events:', len(cluster)*100 / len(n))
76.     print('Min values:', min(cluster, key=lambda x: x[0] + x[1]))
77.     print('Max values:', max(cluster, key=lambda x: x[0] + x[1]))
78.     print('Standard Deviation for x:', (stdev(cluster[:,0])))
79.     print('Standard Deviation for y:', (stdev(cluster[:,1])))
80.     print("\n")
81.
82. # AVERAGE VALUE
```

```
print('Mean x:', np.mean(x1))
print('Mean y:', np.mean(x2))
print("\n")

# CENTROIDS VALUES
print("Centroids:", centroids)

# 03. PLOTTING THE CLUSTERS
for i in range(len(X)):
    plt.plot(X[i][0], X[i][1], colors[labels[i]], markersize=10)
plt.scatter(centroids[:,0], centroids[:,1], marker='.', s=20, linewidths=5, zorder=10, color='k')
plt.xlabel('Goldstein scale')
plt.ylabel('Average tone')
for x1,x2,txt in np.broadcast(x1,x2,n):
    plt.annotate(txt, (x1,x2))

# Range for ten countries (commment if using all countries)
      plt.xlim(0,1.3)
      plt.ylim(-4.5, -1)
      # Range for all countries (commment if using ten countries)
      plt.xlim(-8,8)
      plt.ylim(-7,4)

      plt.show()
```

9. Countries clustered by month

Script '10_countries_clustering_months.py' from GDELT-ANALYSIS repository.

```
from statistics import stdev

import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
from pymongo import MongoClient
from sklearn.cluster import KMeans

# CONNECTION WITH MONGODB
client = MongoClient()
client = MongoClient("mongodb://localhost:27017/")

db = client.GDELT
coll = db.Events

# FIND EVENTS
events = coll.aggregate([{'$match': {'ActionGeo_CountryCode': { '$in': ['BR','CH','IN','IT','PL','RS','SP','TU','UK','US'] }}}, {'$sample': { 'size': 500000 }}], allowDiskUse= True )
data = list(events)
events.close()
df = pd.DataFrame(data)

grouped = df.groupby('MonthYear')
for name, df in grouped:
    # Remove rows with NaN values
    df = df.replace('null', np.nan, regex=True)
    df = df[df['ActionGeo_CountryCode'].notna()]

```

```
    # 01. Goldstein scale and average tone
    df['x'] = pd.to_numeric(df['GoldsteinScale']).values
    df['y'] = pd.to_numeric(df['AvgTone']).values
    df = df.filter(['ActionGeo_CountryCode','x','y'], axis=1)
    df = df.groupby('ActionGeo_CountryCode').mean()

    # MEASURES (INPUTS)
    x1 = np.array(df['x'])
    x2 = np.array(df['y'])
    n = list(df.groupby('ActionGeo_CountryCode').groups.keys())

    # CREATE COORDINATES WITH THE INPUTS (x,y)
    X = np.array(list(zip(x1, x2)))
    X.shape

    # Centroids for ten selected countries (commment if using all countries)
    centroids = np.array([[0.45,-2.95], [0.86,-2.54], [0.95,-
1.55]], np.float64)
    # Centroids for all countries (commment if using ten countries)
    centroids = np.array([[-0.37,-3.61], [0.93, -2.07], [1.61,-
0.62]], np.float64)

    # ASSIGNING NUMBER OF CLUSTERS FOR KMEANS
    kmeans = KMeans(n_clusters=3, init=centroids, n_init=1)
    kmeans = kmeans.fit(X)
    labels = kmeans.predict(X)
    colors=["m.", "r.", "c.", "y.", "b."]

    # 01. SEPARATING VALUES FOR DIFFERENT CLUSTERS
    def groupby(X, labels):
        sidx = labels.argsort(kind='mergesort')
        X_sorted = X[sidx]
        labels_sorted = labels[sidx]

        cut_idx = np.flatnonzero(np.r_[True, labels_sorted[1:] != labels_sorte
d[:-1], True])

        out = [X_sorted[i:j] for i,j in zip(cut_idx[:-1],cut_idx[1:])]
        return out

    result = groupby(X, labels)

    # 02. CALCULATIONS FOR EACH CLUSTER
    for cluster in result:
        for i in range(len(cluster)):
            sums = cluster[i][0] + cluster[i][1]

        print('Percentage of events:', len(cluster)*100 / len(n))
        print('Min values:', min(cluster, key=lambda x: x[0] + x[1]))
        print('Max values:', max(cluster, key=lambda x: x[0] + x[1]))
        print('Standard Deviation for x:', (stdev(cluster[:,0])))
        print('Standard Deviation for y:', (stdev(cluster[:,1])))
        print("\n")

    # AVERAGE VALUE
    print('Mean x:', np.mean(x1))
    print('Mean y:', np.mean(x2))
    print("\n")

    # CENTROIDS VALUES
    print("Centroids:", centroids)

    # 03. PLOTTING THE CLUSTERS
    for i in range(len(X)):
        plt.plot(X[i][0], X[i][1], colors[labels[i]], markersize=10)
```

```
    plt.scatter(centroids[:,0], centroids[:,1], marker='.', s=20, linewidths=5
, zorder=10, color='k')
    plt.xlabel('Goldstein scale')
    plt.ylabel('Average tone')

    for x1,x2,txt in np.broadcast(x1,x2,n):
        plt.annotate(txt, (x1,x2))

    plt.title(name)
    # Range for ten countries (commment if using all countries)
    plt.xlim(0,1.3)
          plt.ylim(-4.5, -1)
          # Range for all countries (commment if using ten countries)
          plt.xlim(-8,8)
          plt.ylim(-7,4)

          plt.show()
```

10. China analysis

Script '11_china_charts.py' from GDELT-ANALYSIS repository.

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
from matplotlib.pyplot import show
from pymongo import MongoClient

client = MongoClient()
client = MongoClient("mongodb://localhost:27017/")

db = client.GDELT
coll = db.Events

# Filter events
events = coll.aggregate([{'$match': {'Actor1Code': 'CHN', 'Actor2CountryCode':
 {
                        '$nin': ['CHN']}}}], allowDiskUse=True)
data = list(events)
events.close()
df = pd.DataFrame(data)

# Remove rows with NaN values
df = df.replace('null', np.nan, regex=True)
df = df[df['Actor1Code'].notna()]

# 01. EVENTS DENSITY EVOLUTION
df['SQLDATE'] = pd.to_datetime(df['SQLDATE'])
df.groupby(pd.Grouper(key='SQLDATE', freq='W'))[
    '_id'].nunique().plot(figsize=(10, 20))
plt.title('EVOLUTION OF EVENTS DENSITY WITH CHINA AS ACTOR 1 CODE')
plt.ylabel('Number of occurrences')
plt.xlabel('Date')
show()

# Filter events
events = coll.aggregate(
```

```
    [{'$match': {'Actor2CountryCode': {'$nin': ['CHN']}}}], allowDiskUse=True)

data = list(events)
events.close()
df = pd.DataFrame(data)

# Remove rows with NaN values
df = df.replace('null', np.nan, regex=True)
df = df[df['Actor2CountryCode'].notna()]

# 02. CORRELATION BETWEEN COUNTRIES AND OTHER COUNTRIES BY MONTH
df.groupby(['MonthYear', 'Actor2CountryCode'])['_id'].nunique().nlargest(
    30).unstack('Actor2CountryCode').plot(kind='bar')
plt.title('CORRELATION BETWEEN CHINA AND OTHER COUNTRIES')
plt.ylabel('Country code')
plt.xlabel('Number of occurrences')
show()
```